**Ensemble-Based Machine Learning Model for Mental Illness Classification on a Mental Health Community Platform**



**Disusun Oleh :**

**Kelompok 9**

**MACHINE LEARNING LG01**

| Arfan Razaqi Rusdi | 2702367170 |
| --- | --- |
| Delo Yohan Sunarto | 2702249316 |
| William Theodorus Wijaya | 2702249316 |
|  |  |

**Universitas Bina Nusantara**

**Kemanggisan**

**2024**

# BAB I

# PENDAHULUAN

## Latar Belakang

Indonesia, khususnya Jakarta, memiliki banyak penyandang disabilitas, termasuk mereka yang mengalami gangguan mental seperti bipolar dan depresi. Data terbaru menunjukkan bahwa terdapat 6.557 orang dengan disabilitas mental-jiwa di Jakarta, menjadikannya salah satu kategori disabilitas terbesar di wilayah tersebut. Meskipun teknologi telah berkembang pesat dan menawarkan peluang besar untuk membantu pengelolaan kondisi mental sehari-hari, banyak penyandang disabilitas mental belum sepenuhnya memanfaatkan potensi teknologi ini. Masalah utamanya adalah kurangnya teknologi yang dirancang secara inklusif dan relevan dengan kebutuhan khusus mereka, sehingga menyebabkan kesulitan dalam mengakses informasi penting, dukungan kesehatan mental, dan partisipasi aktif dalam masyarakat. Stigma sosial dan kurangnya dukungan lingkungan juga memperparah isolasi mereka dari manfaat teknologi.

Berangkat dari kondisi tersebut, riset ini menjadi krusial untuk memahami secara mendalam pengalaman, tantangan, dan kebutuhan spesifik para pengguna dengan gangguan bipolar dan depresi. Dengan menggali wawasan langsung dari individu yang beragam, penelitian ini berupaya mengidentifikasi celah yang ada pada solusi teknologi saat ini. Tujuannya adalah untuk merumuskan desain dan fitur aplikasi yang lebih tepat sasaran, sehingga teknologi yang dikembangkan nantinya tidak hanya sekedar memenuhi kebutuhan mereka, tetapi juga secara efektif mendukung mereka dalam menjalani kehidupan sehari-hari dengan lebih baik dan mandiri.

## Permasalahan

Permasalahan inti yang dihadapi oleh penyandang disabilitas mental adalah sering kali tidak terlayani dengan baik oleh teknologi yang ada. Hal ini disebabkan oleh beberapa faktor, termasuk keterbatasan dalam desain yang inklusif, kurangnya pemahaman mendalam tentang kebutuhan khusus mereka, dan kekurangan fitur yang benar-benar relevan dengan pengalaman hidup mereka. Akibatnya, mereka kesulitan dalam menggunakan teknologi untuk mendukung kesehatan mental, mengakses informasi penting, atau berpartisipasi aktif dalam masyarakat. Stigma sosial dan kurangnya dukungan dari lingkungan sekitar juga memperparah situasi, membuat mereka semakin terisolasi dari manfaat teknologi yang seharusnya dapat membantu mereka.

## Tujuan Penelitian

Mood Bridge merupakan sebuah aplikasi berbasis web yang dirancang untuk mendukung individu dengan disabilitas mental dalam mengekspresikan diri dan memperoleh dukungan emosional. Aplikasi ini menyediakan fitur-fitur utama seperti:

* Meningkatkan kesadaran dan kesejahteraan penyandang disabilitas mental melalui teknologi.
* Memberikan alat digital yang konsisten dan mudah diakses untuk menjaga kesehatan mental.
* Menghilangkan stigma dengan menghadirkan ruang aman berbasis komunitas dan privasi.

## Target audiens

* Penyandang disabilitas mental remaja dan dewasa muda.
* Keluarga atau pengasuh yang mendampingi.
* Organisasi non-profit atau komunitas yang bergerak di bidang kesehatan mental.

# BAB II

# PENDEKATAN & METODOLOGI

## Pendekatan Pembuatan Model Machine Learning

Penelitian ini akan mengadopsi pendekatan riset sekunder secara eksklusif untuk memperoleh pemahaman yang komprehensif mengenai pengalaman, tantangan, dan kebutuhan pengguna dengan disabilitas mental, khususnya penderita bipolar dan depresi. Fokus utama akan diberikan pada studi literatur yang ekstensif. Studi ini akan mencakup pengumpulan dan analisis penelitian sebelumnya mengenai dampak gangguan bipolar terhadap kehidupan sehari-hari, karakteristik, gejala, dan tantangan yang dihadapi penderita. Pendekatan ini akan memberikan landasan teoritis yang kuat dan wawasan dari temuan riset yang telah dipublikasikan.

Dalam konteks pengembangan model Machine Learning untuk proyek ini, yang meskipun didasarkan pada riset sekunder, dapat diarahkan pada masalah klasifikasi, beberapa model yang relevan untuk dipertimbangkan antara lain XGBoost, Logistic Regression, dan Voting Classifier.

* XGBoost

XGBoost menangani klasifikasi berdasarkan fitur-fitur tersebut, menawarkan keseimbangan antara kemampuan pemrosesan teks tingkat lanjut dan performa klasifikasi yang kuat.

* Logistic Regressions  
  Berkemampuan untuk mengidentifikasi hubungan linier antara fitur dan kelas target membuatnya menjadi pilihan yang baik untuk masalah klasifikasi biner atau multinomial yang memiliki asumsi linieritas yang mendasari.
* Voting Classifier  
  Dalam pendekatan ini, model-model seperti CatBoost, XGBoost, dan LightGBM serta output dari mereka, dilatih secara terpisah. Kemudian,

Voting Classifier akan menggabungkan prediksi mereka untuk membuat keputusan klasifikasi akhir. Metode ini bertujuan untuk meningkatkan robusta dan akurasi model secara keseluruhan dengan memanfaatkan kekuatan beragam algoritma dan mengurangi bias dari model tunggal.

## Dataset

Dataset yang digunakan berasal dari [kaggle.com](http://kaggle.com), yang berisi teks berupa pernyataan dari pengguna serta label klasifikasi terkait kondisi kesehatan mental. Label-label ini mewakili berbagai jenis gangguan seperti depresi, bipolar, ADHD, dan lainnya. Dataset ini diproses menjadi DataFrame menggunakan library pandas, dan dilakukan pembagian data menjadi data latih dan data uji dengan perbandingan 80:20 menggunakan train\_test\_split.

## Metode Preprocessing

Preprocessing dilakukan untuk membersihkan teks agar dapat diproses lebih baik oleh model. Langkah-langkah preprocessing meliputi:

* Konversi seluruh teks menjadi huruf kecil (*lowercase*)
* Penghapusan tag HTML
* Penghapusan URL
* Penghapusan tanda baca (*punctuation*)
* Penghapusan angka dan karakter khusus
* Tokenisasi dan penghapusan stopwords menggunakan NLTK
* Representasi teks menggunakan **TF-IDF Vectorizer** agar dapat diproses oleh algoritma machine learning

# BAB III

# IMPLEMENTASI

Aplikasi web ini dibangun dengan arsitektur modern yang memisahkan antara antarmuka pengguna (*frontend*) dan logika bisnis di server (*backend*). Pengguna berinteraksi melalui *frontend* yang dibuat menggunakan **Next.js (T3 Stack)**, memberikan pengalaman yang cepat dan interaktif dengan *styling* dari **Tailwind CSS**. Di sisi *backend*, sistem ditenagai oleh **Golang** dengan *framework* **Gin Engine** untuk menangani permintaan API dan **Gorilla WebSocket** untuk komunikasi *real-time*. Inti dari fungsionalitas aplikasi, yaitu klasifikasi *tweet*, dijalankan oleh model AI *ensemble* (**CatBoost, XGBoost, LightGBM**) yang di-*deploy* secara terpisah menggunakan **FastAPI**. Seluruh data, termasuk data pengguna dan *tweet*, disimpan dalam *database* **PostgreSQL** yang dikelola melalui platform **Supabase**. Untuk keamanan dan manajemen sesi pengguna, sistem otentikasi diintegrasikan menggunakan **Next-Auth**.

## Frontend

## **Next.js (T3 Stack)**: Berperan sebagai kerangka kerja utama untuk membangun antarmuka pengguna (UI). T3 Stack (Next.js, TypeScript, tRPC, Prisma, NextAuth) memastikan pengembangan yang sangat efisien dan aman (*type-safe*). Next.js memungkinkan rendering sisi server (SSR) dan generasi situs statis (SSG) yang membuat aplikasi lebih cepat dan ramah SEO.

## **Tailwind CSS**: Digunakan untuk *styling* atau desain visual aplikasi. Sebagai kerangka kerja CSS *utility-first*, Tailwind mempercepat proses desain dengan memungkinkan developer membangun antarmuka kustom langsung di dalam kode HTML/JSX tanpa perlu menulis CSS kustom.

## Backend

## **Golang**: Bahasa pemrograman yang menjadi otak dari sisi server. Dipilih karena performanya yang sangat tinggi, efisiensi memori, dan kemampuan menangani banyak proses secara bersamaan (*concurrency*), yang ideal untuk aplikasi dengan lalu lintas tinggi.

## **Gin Engine**: Kerangka kerja (*framework*) web untuk Golang. Gin sangat ringan dan cepat, berfungsi untuk menyederhanakan pembuatan rute API (*routing*), menerima permintaan dari *frontend*, dan mengirimkan respons kembali.

## **Gorilla WebSocket**: Library yang digunakan untuk mengimplementasikan komunikasi dua arah secara *real-time* antara *frontend* dan *backend*. Ini berguna untuk fitur seperti notifikasi langsung atau pembaruan data otomatis tanpa perlu me-*refresh* halaman.

## **Arsitektur Perangkat Lunak:** Clean Architecture Konsep Clean Architecture diterapkan untuk menstrukturkan kode di sisi *backend* (Golang) agar lebih terorganisir, mudah diuji (*testable*), dan fleksibel terhadap perubahan. Arsitektur ini memisahkan logika bisnis inti dari detail teknis seperti *framework* web atau *database*. Penerapannya membagi kode ke dalam beberapa lapisan konsentris:

## **Entities (Domain)**: Lapisan terdalam yang berisi objek bisnis inti dan aturan-aturan universal. Contohnya adalah struct untuk User atau Tweet dalam Go. Lapisan ini tidak bergantung pada lapisan lain mana pun.

## **Usecases (Application Business Rules)**: Lapisan ini berisi logika spesifik aplikasi. Misalnya, *usecase* untuk "Mendaftarkan Pengguna" atau "Mengklasifikasikan Tweet". Lapisan ini mendefinisikan *interface* (kontrak) untuk dependensi luar, seperti *repository* untuk akses *database*.

## **Interface Adapters**: Lapisan ini bertindak sebagai jembatan antara *usecase* dan dunia luar. Ini mencakup:

## **Controllers/Handlers**: Fungsi-fungsi Gin yang menerima permintaan HTTP, memvalidasi input, memanggil *usecase* yang sesuai, dan mengembalikan respons HTTP.

## **Repositories**: Implementasi konkret dari *interface* yang didefinisikan di lapisan *usecase*. Di sinilah kode untuk berinteraksi dengan *database* PostgreSQL/Supabase berada.

## **Gateways**: Komponen yang berinteraksi dengan layanan eksternal, seperti melakukan panggilan API ke model FastAPI.

## **Frameworks & Drivers**: Lapisan terluar yang berisi semua detail teknis seperti *framework* Gin itu sendiri, *driver database* PostgreSQL, dan library Gorilla WebSocket. Lapisan ini berfungsi untuk "mengikat" semua komponen menjadi satu kesatuan aplikasi yang berjalan.

## **Keuntungan utama dari pendekatan ini adalah:**

## **Independensi**: Logika bisnis tidak bergantung pada *framework* atau *database*. Anda bisa mengganti Gin dengan *framework* lain atau PostgreSQL dengan MySQL tanpa mengubah kode di lapisan *Use Cases* atau *Entities*.

## **Testabilitas Tinggi**: Setiap lapisan dapat diuji secara terpisah. *Use Case* dapat diuji dengan *mock repository* tanpa memerlukan koneksi *database* sungguhan.

## Kemudahan Perawatan: Kode menjadi lebih mudah dipahami dan dikelola karena adanya pemisahan tanggung jawab yang jelas.

## Database

* **Supabase / PostgreSQL**: **PostgreSQL** adalah sistem *database* relasional yang kuat dan andal untuk menyimpan semua data aplikasi. **Supabase** bertindak sebagai platform *Backend-as-a-Service* (BaaS) yang menyediakan *database* PostgreSQL yang sudah dikelola, serta fitur tambahan seperti otentikasi, *storage*, dan API instan yang mempermudah dan mempercepat pengembangan *backend*.

## AI Model / Deployment

## Ensemble (CatBoost, XGBoost, LightGBM): Ini adalah model *machine learning* untuk melakukan tugas utama, yaitu klasifikasi *tweet*. Menggunakan metode *ensemble* (menggabungkan tiga model *gradient boosting* yang kuat) bertujuan untuk mendapatkan akurasi klasifikasi yang lebih tinggi daripada menggunakan satu model saja.

## FastAPI: Kerangka kerja Python yang digunakan khusus untuk menyajikan (*deploy*) model AI sebagai sebuah API. FastAPI dipilih karena performanya yang sangat cepat dan kemudahannya dalam membuat API, sangat cocok untuk lingkungan produksi *machine learning*. *Backend* Golang akan berkomunikasi dengan API ini untuk mendapatkan hasil prediksi.

# 

# BAB IV

# EVALUASI & HASIL

## **Evaluasi Model**

Setelah dilakukan pelatihan dan pengujian, model Voting Classifier berhasil menunjukkan performa yang sangat baik dalam melakukan klasifikasi gangguan kesehatan mental berdasarkan data teks. Evaluasi dilakukan terhadap data uji menggunakan berbagai metrik performa klasifikasi.

**Label:**

**· Anxiety[0],**

**· Bipolar [1],**

**· Depression [2],**

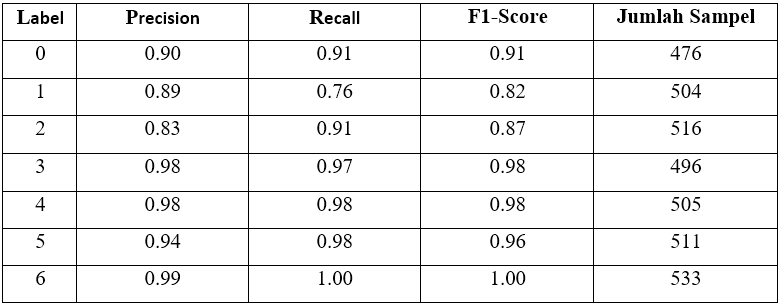
**· Normal [3],**

**· Personality Disorder [4],**

**· Stress [5],**

**· Suicidal [6]**

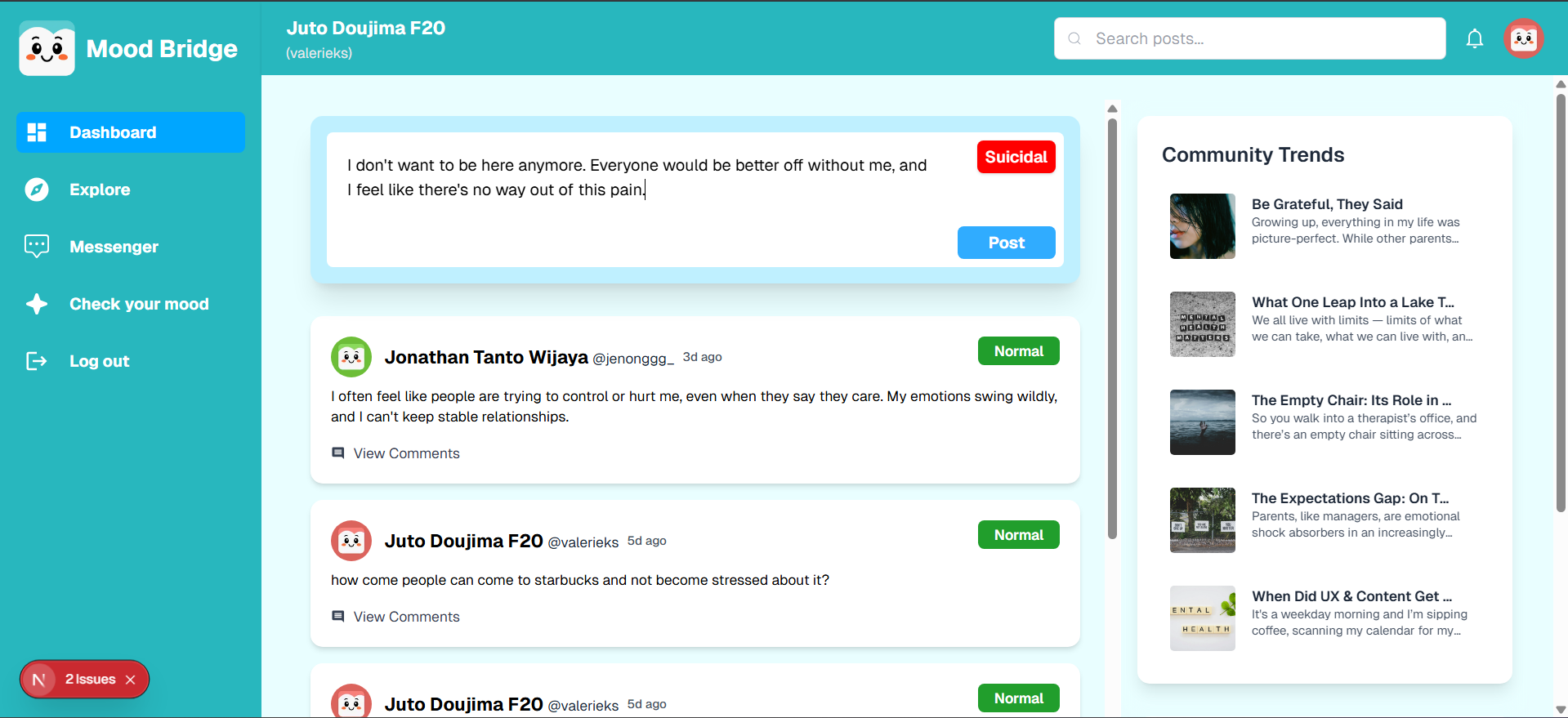
**Evaluation Table:**

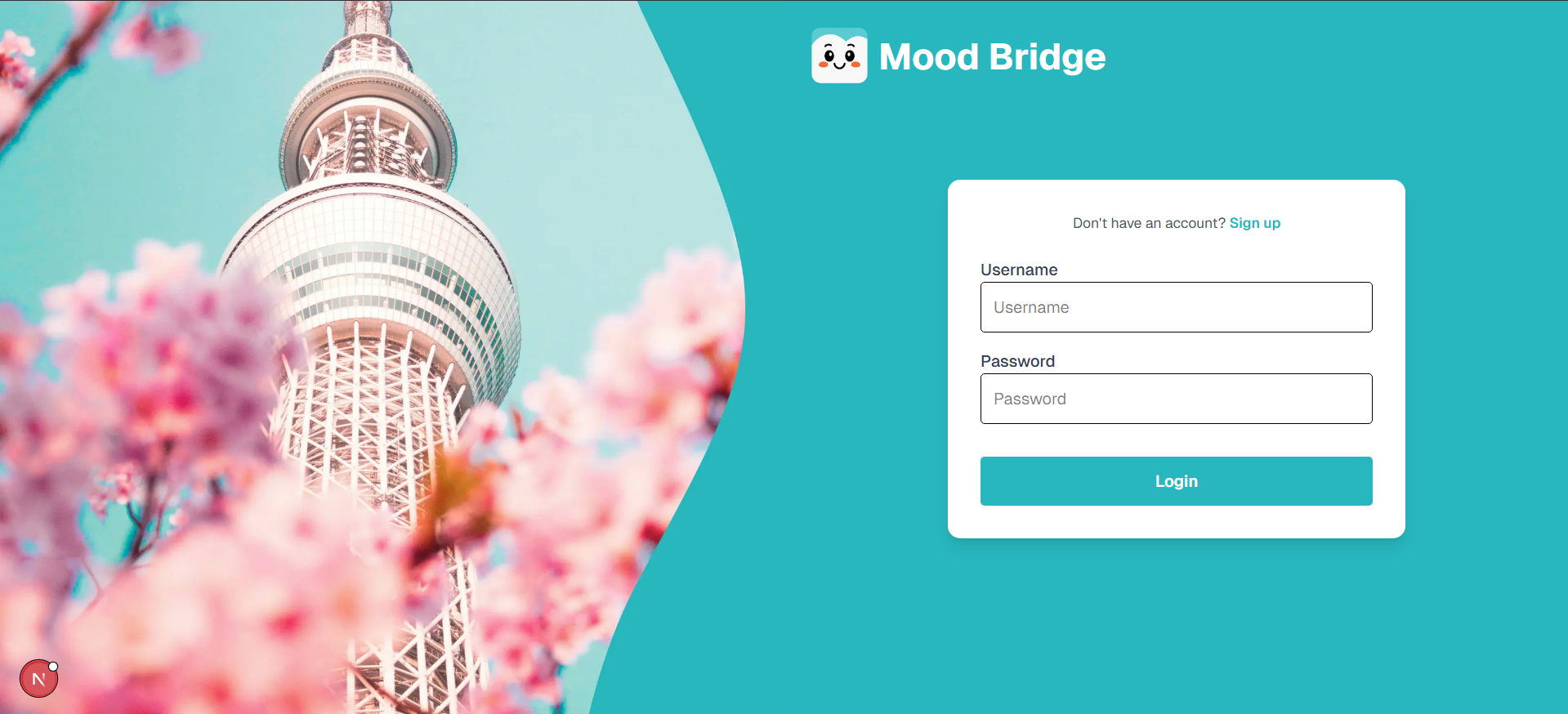


**Confusion Matrix:**

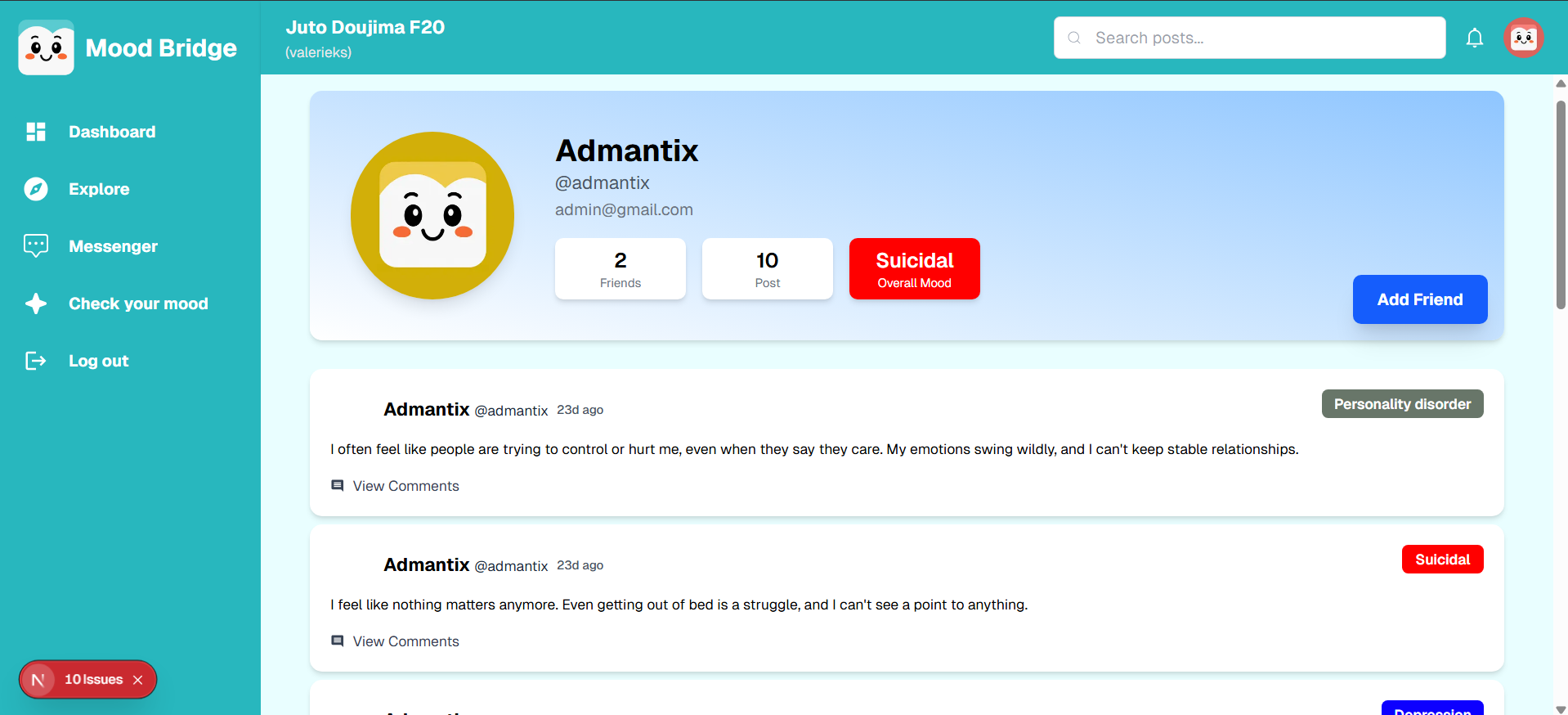
## 

## **Hasil Website**









**BAB V**

**PEMBAGIAN TUGAS KELOMPOK**

| **Pembagian Tugas** | **Nama** |
| --- | --- |
| Model Building & Deployment | William Theodorus Wijaya |
| Backend Developer | William Theodorus Wijaya |
| Frontend Developer | Arfan Razaqi Rusdi & William Theodorus Wijaya |
| UI/UX Designer | Delon Yohan Sunarto |