# CIS4813 Web Applications Programming: Semester Project | Working Prototype / MVP

**1. Student's Registration Information**

- **Full Names & IDs**:
  - Carter Susi: 200649966, cts24d
  - Albert Velazques: 200198435, amv24h
  - Anika Guevarra: 200725744, asg23d
- **Course Name**: COP4813
- **Instructor Name**: Dr Ahsan Abdullah
- **Date of Submission**: 06-22-2025

## 1. Title and Team Members

- **Friend Finder** (Clifton Strengths Assessment)

- **Team Members**:

  - Carter Susi

  - Albert Velazques

  - Anika Guevarra

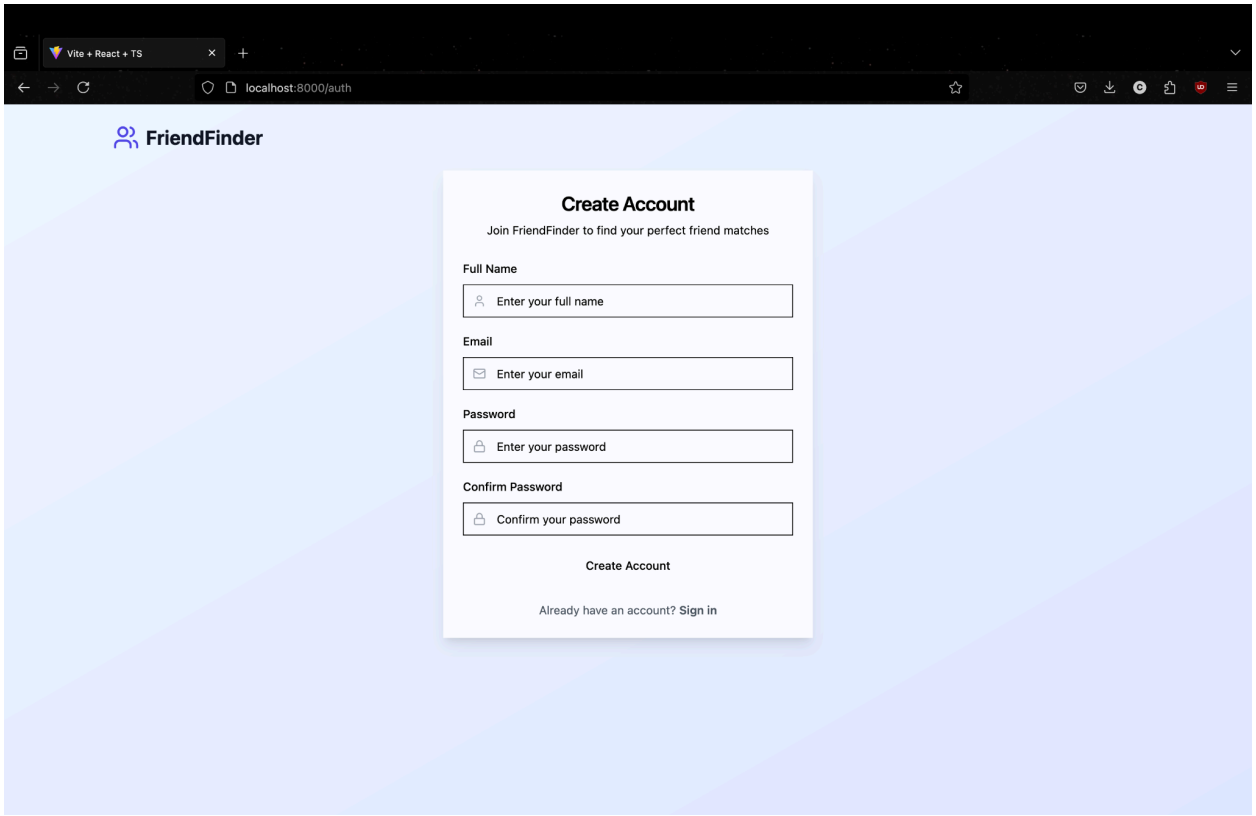
---

# 2. Front-End Form Pages

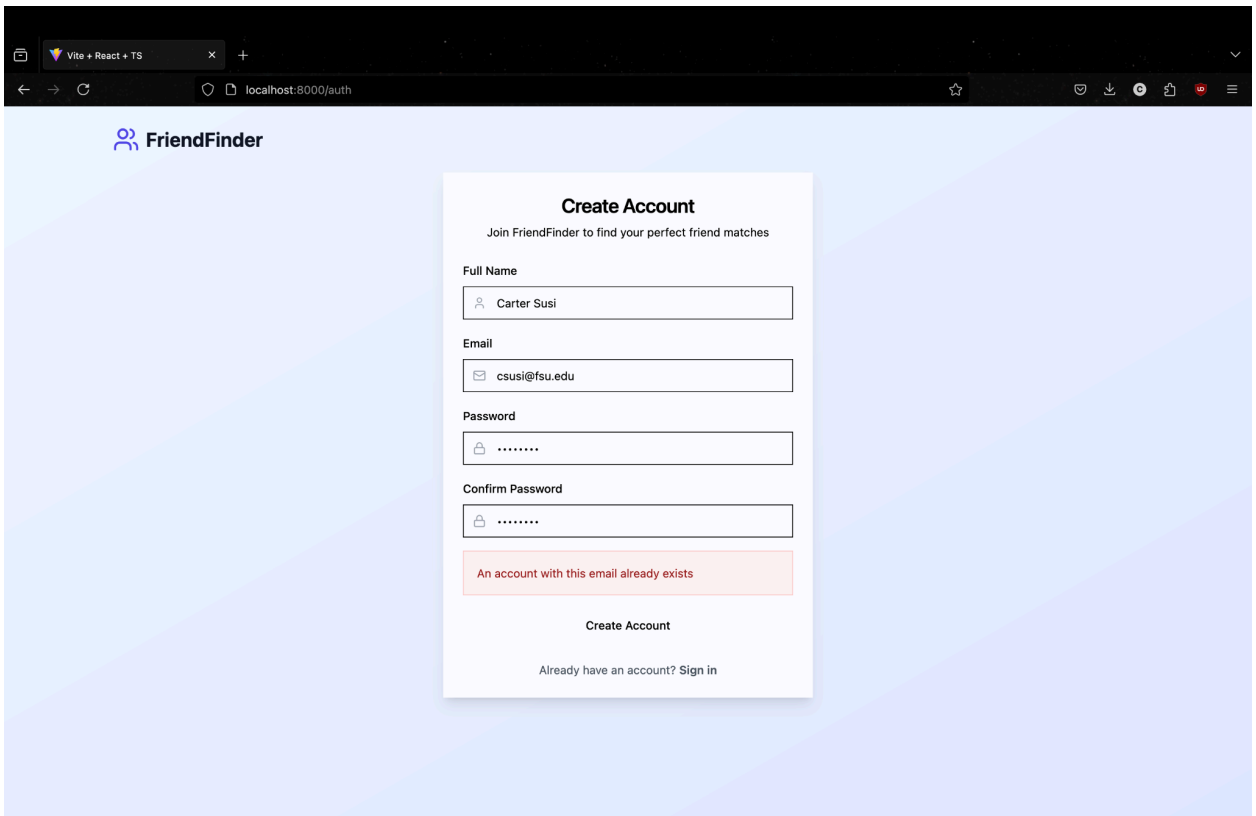
## User Auth: Sign in | Sign up | Session Management

**Basic Flow:** Users can sign in or sign up at "/auth" which will create a session for the user. This allows them to access restricted pages where a session key is required.

## Empty Auth Page:



# Account Recreation:



## 3. End-to-End Flow

### Auth API Schema

```python
# Pydantic models for request/response
class LoginRequest(BaseModel):
    email: EmailStr
    password: str

class SignupRequest(BaseModel):
    name: str
    email: EmailStr
    password: str

class AuthResponse(BaseModel):
    success: bool
    message: Optional[str] = None
    user: Optional[UserResponse] = None
    session: Optional[SessionResponse] = None

@auth_router.post("/login", response_model=AuthResponse)
@auth_router.post("/signup", response_model=AuthResponse)
@auth_router.post("/logout", response_model=AuthResponse)
```

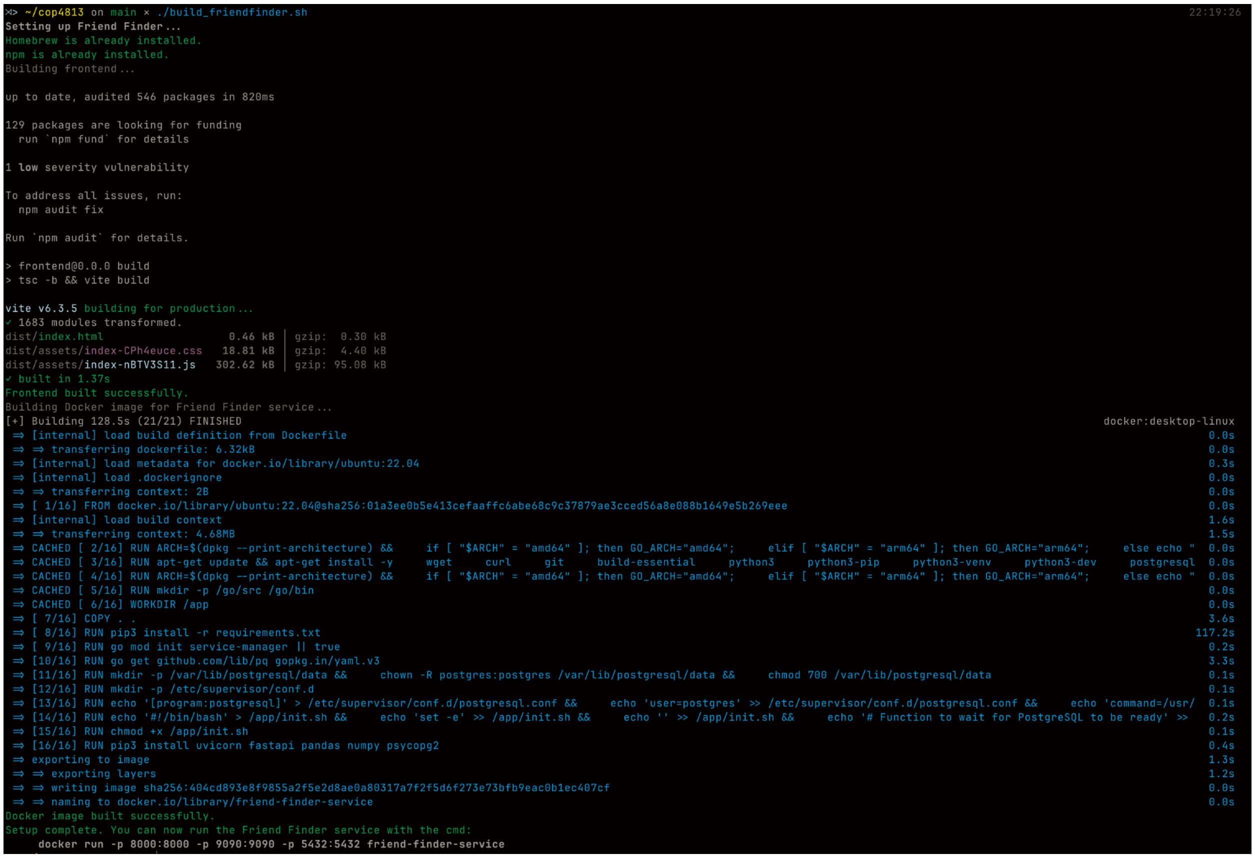
### Sign-Up Logging:

```
● React app available at: http://localhost:8000/
INFO:     127.0.0.1:50874 - "GET /health HTTP/1.1" 200 OK
[SERVICE-MANAGER] 2025/06/22 02:33:23 service-manager.go:329: [PYTHON-STDOUT] INFO:     127.0.0.1:37740 - "GET /health HTTP/1.1" 200 OK
[SERVICE-MANAGER] 2025/06/22 02:33:38 service-manager.go:329: [PYTHON-STDOUT] INFO:     172.17.0.1:63978 - "GET / HTTP/1.1" 200 OK
[SERVICE-MANAGER] 2025/06/22 02:33:38 service-manager.go:329: [PYTHON-STDOUT] INFO:     172.17.0.1:63978 - "GET /assets/index-nBTV3S11.js HTTP/1.1" 304 Not Modified
[SERVICE-MANAGER] 2025/06/22 02:33:38 service-manager.go:329: [PYTHON-STDOUT] INFO:     172.17.0.1:63992 - "GET /assets/index-CPh4euce.css HTTP/1.1" 304 Not Modified
[SERVICE-MANAGER] 2025/06/22 02:33:41 service-manager.go:329: [PYTHON-STDOUT] INFO:     172.17.0.1:63978 - "GET /auth HTTP/1.1" 200 OK
[SERVICE-MANAGER] 2025/06/22 02:33:53 service-manager.go:329: [PYTHON-STDOUT] Connected to PostgreSQL database: friend_finder
User created successfully with ID: 3
Database connection closed
INFO:     172.17.0.1:60508 - "POST /api/auth/signup HTTP/1.1" 200 OK
[SERVICE-MANAGER] 2025/06/22 02:33:53 service-manager.go:329: [PYTHON-STDOUT] INFO:     127.0.0.1:40594 - "GET /health HTTP/1.1" 200 OK
[SERVICE-MANAGER] 2025/06/22 02:34:00 service-manager.go:329: [PYTHON-STDOUT] Connected to PostgreSQL database: friend_finder
Database connection closed
INFO:     172.17.0.1:56470 - "POST /api/auth/signup HTTP/1.1" 200 OK
[SERVICE-MANAGER] 2025/06/22 02:34:23 service-manager.go:329: [PYTHON-STDOUT] INFO:     127.0.0.1:47644 - "GET /health HTTP/1.1" 200 OK
[SERVICE-MANAGER] 2025/06/22 02:34:24 service-manager.go:329: [PYTHON-STDOUT] Connected to PostgreSQL database: friend_finder
Database connection closed
INFO:     172.17.0.1:59880 - "POST /api/auth/login HTTP/1.1" 200 OK
[SERVICE-MANAGER] 2025/06/22 02:34:25 service-manager.go:329: [PYTHON-STDOUT] Connected to PostgreSQL database: friend_finder
Database connection closed
INFO:     172.17.0.1:59880 - "POST /api/auth/login HTTP/1.1" 200 OK
[SERVICE-MANAGER] 2025/06/22 02:34:53 service-manager.go:329: [PYTHON-STDOUT] INFO:     127.0.0.1:47702 - "GET /health HTTP/1.1" 200 OK
```

# 4. Hosting Local / Cloud

## Hosting: Local

**Docker(Build):** The program is built via build script in an Ubuntu docker image.



```
⋊> ~/cop4813 on main × ./build_friendfinder.sh                                                                                                                                   22:19:26
Setting up Friend Finder...
Homebrew is already installed.
npm is already installed.
Building frontend...

up to date, audited 546 packages in 820ms

129 packages are looking for funding
  run `npm fund` for details

1 low severity vulnerability

To address all issues, run:
  npm audit fix

Run `npm audit` for details.

> frontend@0.0.0 build
> tsc -b && vite build

vite v6.3.5 building for production...
✓ 1683 modules transformed.
dist/index.html                   0.46 kB │ gzip:  0.30 kB
dist/assets/index-CPh4euce.css   18.81 kB │ gzip:  4.40 kB
dist/assets/index-nBTV3S11.js   302.62 kB │ gzip: 95.08 kB
✓ built in 1.37s
Frontend built successfully.
Building Docker image for Friend Finder service...
[+] Building 128.5s (21/21) FINISHED                                                                                                                                  docker:desktop-linux
 ⇒ [internal] load build definition from Dockerfile                                                                                                                                   0.0s
 ⇒ ⇒ transferring dockerfile: 6.32kB                                                                                                                                                  0.0s
 ⇒ [internal] load metadata for docker.io/library/ubuntu:22.04                                                                                                                        0.3s
 ⇒ [internal] load .dockerignore                                                                                                                                                      0.0s
 ⇒ ⇒ transferring context: 2B                                                                                                                                                         0.0s
 ⇒ [ 1/16] FROM docker.io/library/ubuntu:22.04@sha256:01a3ee0b5e413cefaaffc6abe68c9c37879ae3cced56a8e088b1649e5b269eee                                                                 0.0s
 ⇒ [internal] load build context                                                                                                                                                      1.6s
 ⇒ ⇒ transferring context: 4.68MB                                                                                                                                                     1.5s
 ⇒ CACHED [ 2/16] RUN ARCH=$(dpkg --print-architecture) &&     if [ "$ARCH" = "amd64" ]; then GO_ARCH="amd64";     elif [ "$ARCH" = "arm64" ]; then GO_ARCH="arm64";     else echo "  0.0s
 ⇒ CACHED [ 3/16] RUN apt-get update && apt-get install -y     wget     curl     git     build-essential     python3     python3-pip     python3-venv     python3-dev     postgresql  0.0s
 ⇒ CACHED [ 4/16] RUN ARCH=$(dpkg --print-architecture) &&     if [ "$ARCH" = "amd64" ]; then GO_ARCH="amd64";     elif [ "$ARCH" = "arm64" ]; then GO_ARCH="arm64";     else echo "  0.0s
 ⇒ CACHED [ 5/16] RUN mkdir -p /go/src /go/bin                                                                                                                                        0.0s
 ⇒ CACHED [ 6/16] WORKDIR /app                                                                                                                                                        0.0s
 ⇒ [ 7/16] COPY . .                                                                                                                                                                   3.6s
 ⇒ [ 8/16] RUN pip3 install -r requirements.txt                                                                                                                                     117.2s
 ⇒ [ 9/16] RUN go mod init service-manager || true                                                                                                                                    0.2s
 ⇒ [10/16] RUN go get github.com/lib/pq gopkg.in/yaml.v3                                                                                                                              3.3s
 ⇒ [11/16] RUN mkdir -p /var/lib/postgresql/data &&     chown -R postgres:postgres /var/lib/postgresql/data &&     chmod 700 /var/lib/postgresql/data                                 0.1s
 ⇒ [12/16] RUN mkdir -p /etc/supervisor/conf.d                                                                                                                                        0.1s
 ⇒ [13/16] RUN echo '[program:postgresql]' > /etc/supervisor/conf.d/postgresql.conf &&     echo 'user=postgres' >> /etc/supervisor/conf.d/postgresql.conf &&     echo 'command=/usr/  0.1s
 ⇒ [14/16] RUN echo '#!/bin/bash' > /app/init.sh &&     echo 'set -e' >> /app/init.sh &&     echo '' >> /app/init.sh &&     echo '# Function to wait for PostgreSQL to be ready' >>   0.2s
 ⇒ [15/16] RUN chmod +x /app/init.sh                                                                                                                                                  0.1s
 ⇒ [16/16] RUN pip3 install uvicorn fastapi pandas numpy psycopg2                                                                                                                     0.4s
 ⇒ exporting to image                                                                                                                                                                 1.3s
 ⇒ ⇒ exporting layers                                                                                                                                                                 1.2s
 ⇒ ⇒ writing image sha256:404cd893e8f9855a2f5e2d8ae0a80317a7f2f5d6f273e73bfb9eac0b1ec407cf                                                                                            0.0s
 ⇒ ⇒ naming to docker.io/library/friend-finder-service                                                                                                                                0.0s
Docker image built successfully.
Setup complete. You can now run the Friend Finder service with the cmd:
     docker run -p 8000:8000 -p 9090:9090 -p 5432:5432 friend-finder-service
```

**Docker(Run):** The docker image runs the service manager.

```
    /usr/lib/postgresql/14/bin/pg_ctl -D /var/lib/postgresql/data -l logfile start

Starting PostgreSQL...
Waiting for PostgreSQL to start...
Waiting for PostgreSQL... (1/30)
PostgreSQL is ready!
Setting up database...
Running database initialization...
Connected to PostgreSQL database: friend_finder

=== Initializing PostgreSQL Database Tables ===
All tables created successfully
Default roles and permissions created

=== PostgreSQL Database System Initialized ===

=== Creating Sample Users ===
User created successfully with ID: 1
User created successfully with ID: 2

=== Creating Sessions ===
Session created for user 1: 0b1d4b03-ebfc-4f4f-a691-d8dbfee9378d
Session created for user 2: 1fe2feb0-14cc-49c6-8147-e3775688b15d

=== Testing Authentication ===
Authentication successful for user: johndoe

=== Assigning Roles ===
Admin role assigned to user 1

=== Database Statistics ===
users: 2 records
user_sessions: 2 records
roles: 4 records
permissions: 8 records
friends: 0 records
results: 0 records
posts: 0 records
user_security_logs: 4 records

=== Cleaning Up ===
Cleaned up 0 expired sessions
Database connection closed
Building Go service manager...
Starting Go service manager...
[SERVICE-MANAGER] 2025/06/22 02:32:50 service-manager.go:147: Starting Service Manager...
[SERVICE-MANAGER] 2025/06/22 02:32:50 service-manager.go:198: Attempting to connect to database: friend_finder
[SERVICE-MANAGER] 2025/06/22 02:32:50 service-manager.go:215: Database connection established
[SERVICE-MANAGER] 2025/06/22 02:32:50 service-manager.go:169: Service Manager started successfully
[SERVICE-MANAGER] 2025/06/22 02:32:50 service-manager.go:224: Starting Python server: python3 server.py on port 8000
[SERVICE-MANAGER] 2025/06/22 02:32:50 service-manager.go:339: Starting health check server on port 9090
[SERVICE-MANAGER] 2025/06/22 02:32:50 service-manager.go:381: Starting database monitor
[SERVICE-MANAGER] 2025/06/22 02:32:50 service-manager.go:262: Python server started with PID: 1949
[SERVICE-MANAGER] 2025/06/22 02:32:51 service-manager.go:329: [PYTHON-STDERR] INFO:     Started server process [1949]
INFO:     Waiting for application startup.
[SERVICE-MANAGER] 2025/06/22 02:32:51 service-manager.go:329: [PYTHON-STDERR] INFO:     Application startup complete.
[SERVICE-MANAGER] 2025/06/22 02:32:51 service-manager.go:329: [PYTHON-STDERR] INFO:     Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
[SERVICE-MANAGER] 2025/06/22 02:32:53 service-manager.go:329: [PYTHON-STDOUT] 🚀 Starting FastAPI server...
📡 API available at: http://localhost:8000/api/
🌐 React app available at: http://localhost:8000/
INFO:     127.0.0.1:50874 - "GET /health HTTP/1.1" 200 OK
```

**Service Manager:** The server process is spawned from a go service manager to read back exit codes and respond accordingly. The service manager also periodically checks the health and database.

```go
type ServiceManager struct {
    config    *Config
    pythonCmd *exec.Cmd
    db        *sql.DB
    logger    *log.Logger
    shutdown  chan os.Signal
    wg        sync.WaitGroup
    ctx       context.Context
    cancel    context.CancelFunc
}

// Start starts all services
func (sm *ServiceManager) Start() error {
    sm.logger.Println("Starting Service Manager...")

    // Initialize database connection
    if err := sm.initDatabase(); err != nil {
        return fmt.Errorf("failed to initialize database: %w", err)
    }

    // Start database monitor
    sm.wg.Add(1)
    go sm.runDatabaseMonitor()

    // Start health check server (separate from Python server)
    sm.wg.Add(1)
    go sm.runHealthCheckServer()

    // Start web server
    sm.wg.Add(1)
    go sm.runWebServer()

    // Wait for shutdown signal
    go sm.waitForShutdown()

    sm.logger.Println("Service Manager started successfully")
    return nil
}
```