🧠 O que você está propondo:

Você quer prever **quantas horas por dia um usuário usaria o app**, **logo após o cadastro**, **sem precisar de histórico de pings**. Isso é:

Input:

- `age`
- `gender`
- `numberofkids`

Output (previsto pelo modelo):

- `onlinehours` (esperado por dia)

☑️ É viável?

Sim, é viável. Tecnicamente, é só você treinar um modelo de **regressão supervisionada** com:

- Features: `age`, `gender`, `numberofkids`
- Target: `onlinehours` (média por dia calculada a partir do `pings.csv` + `test.csv`)

Claro que o modelo não vai ser superpreciso, pois:

- Está ignorando comportamento real (pings).
- Vai depender só de dados demográficos.

Mas para seu objetivo — incluir ML leve em um sistema simples — é uma ótima ideia.

📦 Pipeline simples que você pode usar:

1. **Pré-processar dados**
    - Agrupar os `pings.csv` por usuário e por dia → calcular `onlinehours`
    - Juntar isso com os dados demográficos do `customers.csv`
    - Calcular a média de `onlinehours` por usuário
    - Criar um `DataFrame final` com: `id, gender, age, numberofkids, mean_onlinehours`
2. **Treinar um modelo**
    - Scikit-learn: `LinearRegression`, `RandomForestRegressor` ou `XGBoostRegressor`
    - Validar com RMSE
3. **Exportar o modelo**
    - Usar `joblib` ou `pickle` para salvar
4. **FastAPI**
    - Criar endpoint `/predict-onlinehours`
    - Receber JSON com `gender`, `age`, `numberofkids`

- Rodar predição e retornar estimativa

5. **Frontend React**
   - Após o usuário preencher o formulário de cadastro, exibir:

     > "Usuários como você costumam usar o app por cerca de **3,2 horas por dia**."

🎁 Exemplo prático:

```
POST /predict-onlinehours

{
  "age": 35,
  "gender": "female",
  "numberofkids": 2
}
```

Resposta do FastAPI:

```
{
  "estimated_online_hours": 2.9
}
```

🚀 Resultado: ML útil, simples e leve

- Sem complicar o frontend.
- Com valor real (preditivo e interpretável).
- Bom para portfólio, mostrando que você entende ML aplicado a produto.
