PRESENTATION

DASH By Plotly

ELIE BEYELER KYLIAN DESCHAMPS KADIR ERAMIL

Framework Python
Dashboards Analytiques Interactifs

OBJECTIFS DE LA PRESENTATION

I - COMPRENDRE CE QU'EST DASH ET SON RÔLE DANS LA CONCEPTION D'INTERFACE

II - IDENTIFIER LES AVANTAGES/INCONVÉNIENTS FACE AUX TECHNOLOGIES CONCURRENTES

III - DÉCOUVRIR LES ACTEURS ET OUTILS LIÉS À DASH

IV - ÉTUDIER LES BASES DE SON FONCTIONNEMENT (LAYOUT, CALLBACKS)

QU'EST-CE QUE DASH ?

UN FRAMEWORK

Framework open-source créé par Plotly

TECHNOLOGIE

Utiliser Python pour éviter JavaScript

CONSTRUIT SUR...

Flask (backend)

React.js (frontend)

Plotly.js (graphiques)

SPÉCIALEMENT CONÇU POUR :

Tableaux de bord analytiques

Interfaces web interactives pour la science des données

Prototypage rapide d'applications

Mettre en avant Plotly

CIBLE APPLICATIVE

TYPE D'APPLICATIONS :

- Web (desktop-first, responsive possible pour mobile)
- Dashboards internes, visualisations, outils collaboratifs

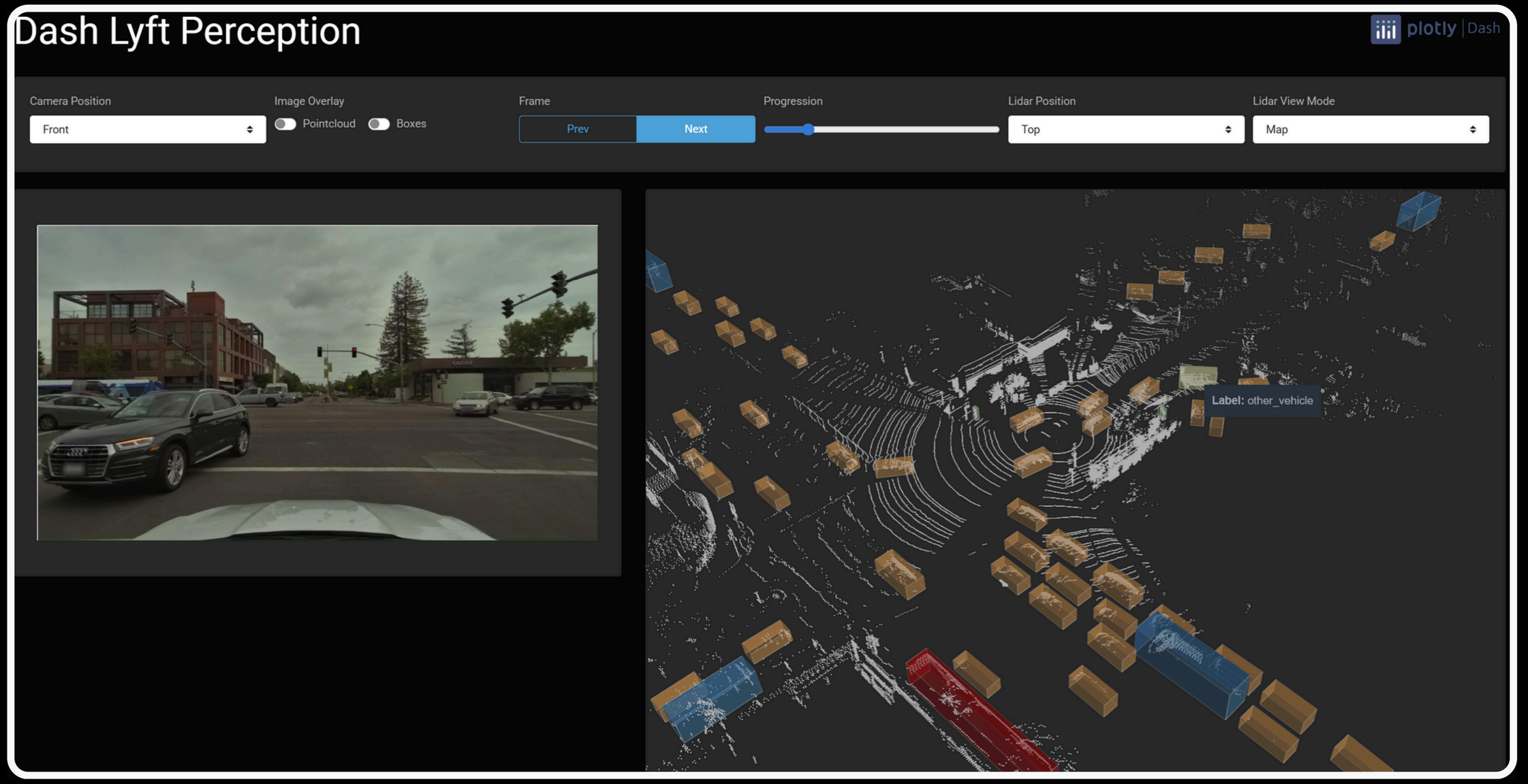
DOMAINES :

- Finance (suivi marché)
- Industrie
- Recherche scientifique
- Startups

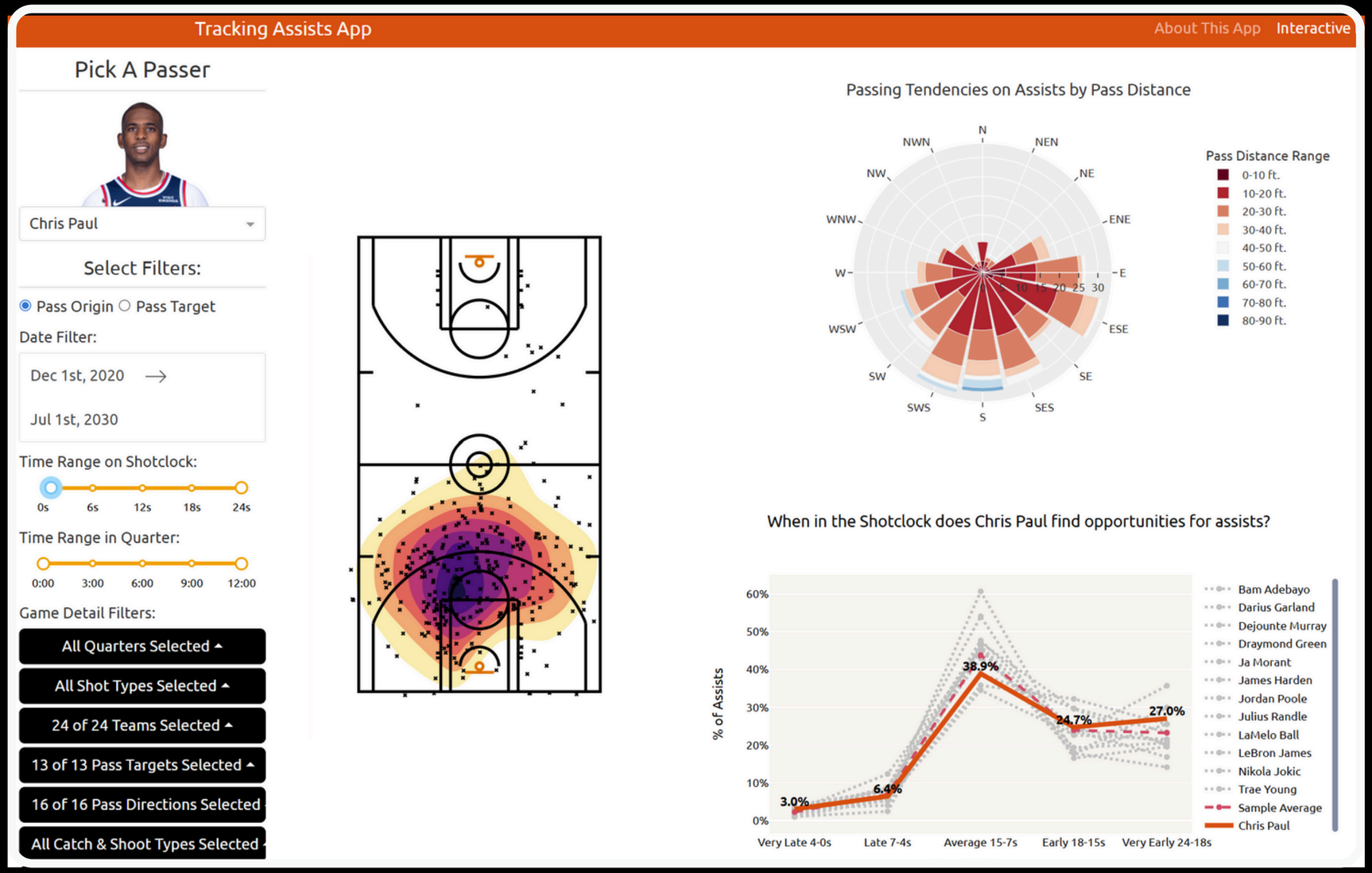
ENTREPRISES :

- S&P Global / Santander
- Colgate / Shell
- Biological and Chemical Research Centre

EXEMPLES CONCRETS 1/4



EXEMPLES CONCRETS 2/4



EXEMPLES CONCRETS 3/4

EXEMPLES CONCRETS 4/4

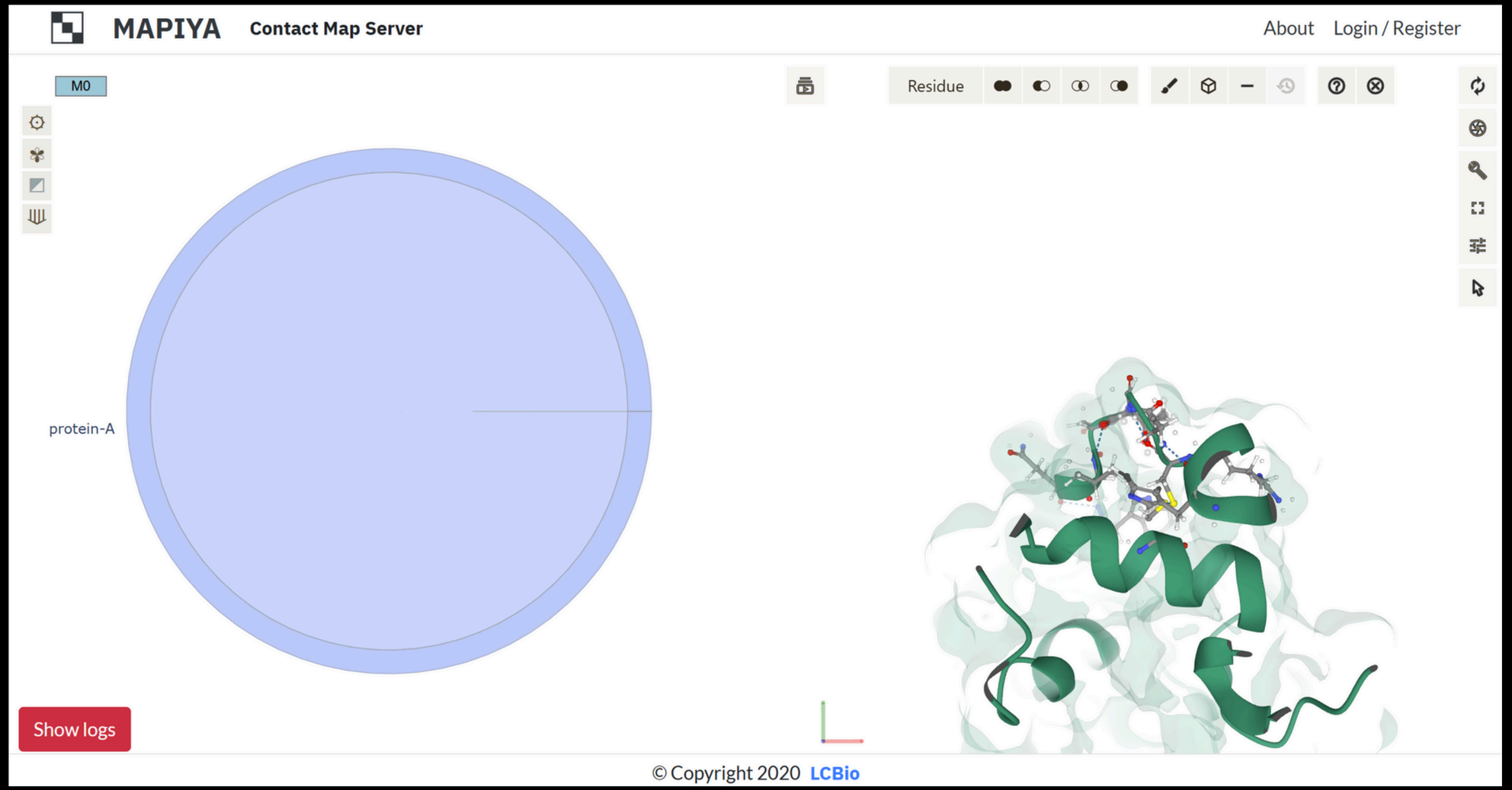


TABLEAU COMPARATIF

Technologie	Points forts	Limites
Dash	Complet, flexible, scalable	Callbacks verbeux
Streamlit	Rapide, intuitif	Peu flexible
Gradio	Complet	Code très long
Panel	Orienté dashboard	Moins populaire
Shiny (Py)	Réactivité simple	Moins sophistiqué

ACTEURS DU MARCHÉ

- Plotly (éditeur de Dash)
- Communauté open-source active (GitHub, forum Plotly)
- Universités / entreprises data-driven
- Dash Enterprise → solution commerciale pour déploiement en entreprise

OUTILS ASSOCIÉS À DASH

LIBRAIRIES PYTHON COMPLÉMENTAIRES :

dash-bootstrap-components → stylisation avec Bootstrap

Pandas pour manipulation de données

Plotly pour visualisation interactive

OUTILS DE DEVELOPPEMENT :

IDE (VSCode, PyCharm)
Jupyter Notebook pour prototypage

DÉPLOIEMENT :

Docker, Render, Heroku, Dash Enterprise

ARCHITECTURE DE DASH

COMPOSANTS PRINCIPAUX :

dash.html : éléments HTML

dash.dcc : composants interactifs (graphes, sliders, dropdowns)

LAYOUT :

Définit la structure de l'affichage de la page

CALLBACKS :

Action utilisateur :
Input
State

Retour pour affichage :
output

EXEMPLE : LAYOUT SIMPLE

```
app.layout = html.Div([
    html.H1("Hello Dash"),
    dcc.Dropdown(
        id=”dropdown-region”,
        options= options_region,
        value =”Toutes”
])
```

- Structure hiérarchique
- Utilisation de composants imitant des balise html ou pour des éléments interactifs

EXEMPLE : CALLBACK SIMPLE

```
@app.callback(
    Output("graph-ventes", "figure"),
    Input("dropdown-region", "value")
)
def update_graph(region):
# Code ...
    return fig
```

- Callback :

→ Output :Renvoyer dans l’affichage

→ Input : Elément interactif et valeur récupéré

- Fonction associé au callback :

→ Utilise les inputs/ génère les outputs

Réactivité : chaque saisie → mise à jour automatique.

GESTION DU STATE

```
@app.callback(
    Output('mon_message', 'children'),
    Input('mon_bouton', 'n_clicks'),
    State('mon_nom', 'value')
)
def dire_bonjour(n_clicks, nom):
    if n_clicks > 0:
        return f"Bonjour {nom} !"
    return " "
```

- State permet de récupérer une valeur sans déclencher un callback en continu.
- Exemple : Champs texte d'un formulaire avant de valider le sélection avec un bouton envoyer.

✅ ID lien composants et callbacks
⚠️ Pour la fonction seul l'ordre compte

MULTIPAGE APPS

```
@app.callback(
Output('page-content', 'children'),
 [Input('url', 'pathname')]
)
def display_page(pathname):
    if pathname == '/informations':
        return page_informations()
```

- Organisation du projet sur une page ou gestion de rooting pour du multi-pages.
- Optimisation des ressource pour afficher uniquement une page.
- Possibilité de fragmenter le code
- ⚠️Ne pas séparer les callbacks d’une page

AVANTAGES DE DASH

- Principalement Python → pas besoin d’utiliser une autre technologie
- Intégration native avec Plotly
- Hautement personnalisable (CSS/JS si besoin)
- Compatible avec Data Science & ML (scikit-learn, TensorFlow, etc.)
- Intégration possible de Dash dans du HTML, JS, SalesForce(CRM)
- Open-source et actif

INCONVENIENTS DE DASH

- Callbacks peuvent devenir nombreux → code verbeux
- Scalabilité : nécessite organisation rigoureuse et optimisations
- Moins adapté aux applications avec animations lourdes
- Courbe d’apprentissage pour gros projets

CONCLUSION

- Dash = outil pour la création de dashboard en python
- Points forts : simplicité et environnement Python, intégration de plusieurs bibliothèques, intégration possibles dans d’autres environnements
- Points faibles : callbacks nombreux, optimisation nécessaire, limitation pouvant amener à compléter par du JS.

RÉFÉRENCES BIBLIOGRAPHIQUES

- Docs officielles : https://dash.plotly.com
- Communauté : forum Plotly, GitHub Dash
- Exemples : https://plotly.com/examples/
- Articles académiques : “Dashboards for data science”

MERCI
NOUS ALLONS PASSER AU TP
HTTPS://GITHUB.COM/FINA49/PROJET_IHM/TREE/TP