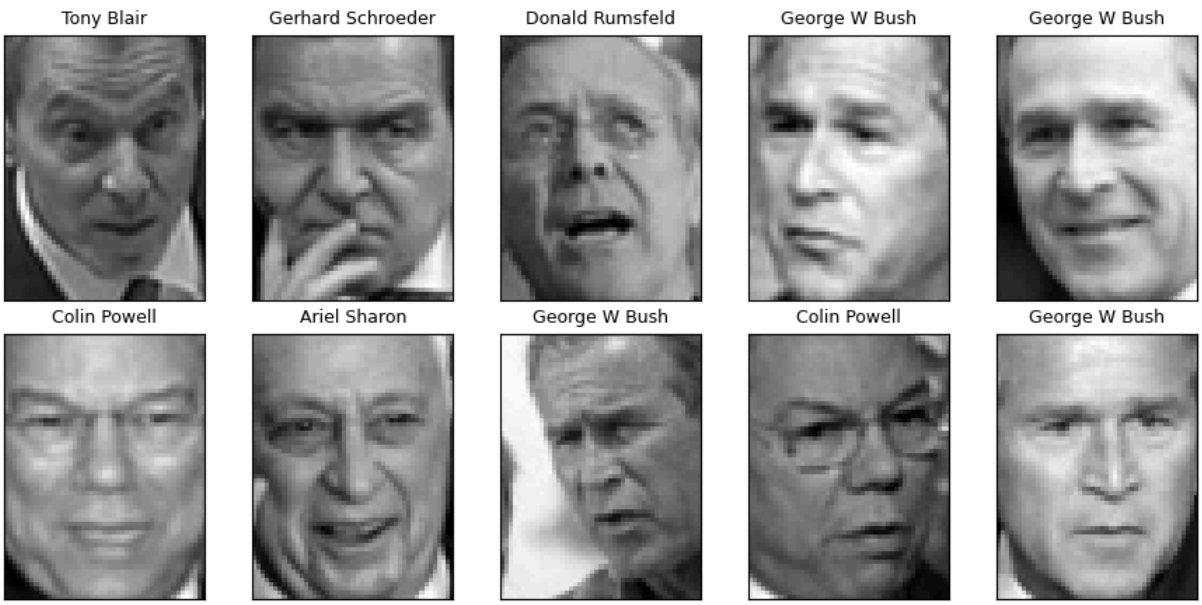
In [86]:
```
import warnings
warnings.filterwarnings('ignore')

# ==================================
# 5-FOLD CV on LFW (Eigenfaces)
# ==================================
import numpy as np
import matplotlib.pyplot as plt
# Labeled Faces in the Wild
from sklearn.datasets import fetch_lfw_people
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.pipeline import Pipeline
from sklearn.svm import SVC

# ---- 1) Load high-dimensional data ----
faces = fetch_lfw_people(min_faces_per_person=50)
X = faces.data
y = faces.target
target_names = faces.target_names
h, w = faces.images.shape[1], faces.images.shape[2]

print("Shape (samples, features):", X.shape, "| classes:", len(np.unique(y)))
print("Classes:", list(target_names))
#print(faces.DESCR)
```

```
Shape (samples, features): (1560, 2914) | classes: 12
Classes: [np.str_('Ariel Sharon'), np.str_('Colin Powell'), np.str_('Donald
Rumsfeld'), np.str_('George W Bush'), np.str_('Gerhard Schroeder'), np.str_
('Hugo Chavez'), np.str_('Jacques Chirac'), np.str_('Jean Chretien'), np.str
_('John Ashcroft'), np.str_('Junichiro Koizumi'), np.str_('Serena William
s'), np.str_('Tony Blair')]
```

In [87]:
```
# --- Display the first 10 face images ---
def show_faces(images, labels, names, h, w, n_row=2, n_col=5):
    plt.figure(figsize=(1.8 * n_col, 2.4 * n_row))
    for i in range(n_row * n_col):
        plt.subplot(n_row, n_col, i + 1)
        plt.imshow(images[i], cmap=plt.cm.gray)
        plt.title(names[labels[i]], size=9)
        plt.xticks(()); plt.yticks(())
    plt.suptitle("First 10 Faces from LFW", size=14)
    plt.tight_layout()
    plt.show()

show_faces(faces.images, y, target_names, h, w)
```



In [88]:
```
# ----- Pipelines -----
pipe_no_pca = Pipeline([
    ("scaler", StandardScaler()),
    ("svc", SVC(kernel="rbf", class_weight="balanced", random_state=42))
])
# Always recommended for LFW class_weight="balanced", more robust on imbalar

pipe_pca = Pipeline([
    ("scaler", StandardScaler()),
    ("pca", PCA(random_state=42)),  # n_components tuned below
    ("svc", SVC(kernel="rbf", class_weight="balanced", random_state=42))
])

# ----- Parameter grids -----
param_no_pca = {
    "svc__C": [0.5, 1, 2, 5, 10]
}

param_pca = {
    "pca__n_components": [75, 100, 150, 200, 300],
    # try with and without whitening; can remove if too slow
    "pca__whiten": [False, True],
    "svc__C": [0.5, 1, 2, 5, 10]
}
```

In [89]:
```
# ----- CV -----
cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)


# ----- Grid searches -----
gs_no_pca = GridSearchCV(
    pipe_no_pca, param_no_pca, cv=cv, scoring="accuracy", n_jobs=-1, verbose
)
gs_pca = GridSearchCV(
    pipe_pca, param_pca, cv=cv, scoring="accuracy", n_jobs=-1, verbose=0
```

```
)

gs_no_pca.fit(X, y)
gs_pca.fit(X, y)

print("\n=== RBF SVM (No PCA) ===")
print(f"Best CV acc: {gs_no_pca.best_score_:.4f}")
print("Best params:", gs_no_pca.best_params_)

print("\n=== RBF SVM + PCA ===")
print(f"Best CV acc: {gs_pca.best_score_:.4f}")
print("Best params:", gs_pca.best_params_)
```

```
=== RBF SVM (No PCA) ===
Best CV acc: 0.7929
Best params: {'svc__C': 10}

=== RBF SVM + PCA ===
Best CV acc: 0.8308
Best params: {'pca__n_components': 75, 'pca__whiten': True, 'svc__C': 1}
```

In [ ]: