Below is an **updated 10-minute presentation** speech that **explicitly** covers each of the instructor's requirements:

1. Present your **problem**.
2. Explain **why/how** it meets the **project requirements**.
3. Provide **sufficient details** (methods, network structure, training, data generation, feature vector usage, novelty detection approach).
4. Make a **live demo** plan.
5. Discuss **possible improvements**.

Use this script as a guide during your talk, adjusting timing and wording as you prefer.

Slide 1: Introduction & Problem Statement (≈1 minute)

> **"Hello, everyone. My name is ___, and today I'll present **'Novelty Detection in Surface Defects with CNN Features, PCA, and K-Means'**.
>
> **Problem**: In our manufacturing plant, a **camera** scans metal surfaces for **5 known defects**—like Crazing or Patches. We discovered a **new** defect type the old system didn't recognize. Our goal: automatically detect and label that new class, so technicians are alerted to a novel fault.
>
> **Why does this matter**? Because real-world processes evolve—**new** defect types appear over time, and we want a system that can highlight "unknown" anomalies for further inspection."*

Slide 2: How It Meets Project Requirements (≈1 minute)

> 1. **Data Analysis**: We do **PCA** on 4096-dimensional CNN features (fulfilling "Data analysis" with SVD).
> 2. **Clustering**: We apply **K-Means** for both base data (k=6) and new data (k=2).
> 3. **Image Processing**: We do **histogram equalization** and **resizing** for each image (fulfilling "Image processing").
> 4. **ANN**: We use **VGG16** (a CNN) to extract features.
>
> **Hence** we cover at least three methods from the list: PCA, K-Means, CNN-based feature extraction, plus image preprocessing."*

Slide 3: Data & Generation (≈1 minute)

> **"We used a subset of the **NEU dataset—1,500** images from 5 known classes, each image is 200×200 grayscale. We also hold out a batch of **40** images containing 20 from a known class and 20 truly new.
>
> **Preprocessing**:
>
> - **Histogram Equalization** to spread intensities and enhance contrast;
> - **Resize** to 224×224 because that's the input size for VGG16.

> Then we store them for the next steps."*

Slide 4: Methods & Network Structure (≈2 minutes)

> "We use **VGG16**, a convolutional neural network architecture pretrained on ImageNet. Specifically, we only do **feature extraction**—we do not retrain it.
>
> - VGG16 has 13 convolution layers + 3 fully-connected layers. We take the **fc1** output, which is a 4096-dimensional vector describing each image's high-level features.
>
> **Why fc1**? It's a widely used representation from VGG16 that often performs well in downstream tasks.
>
> After that, we do:
>
> 1. **PCA** from 4096→50 dimensions, using **SVD=full** so it handles the entire dataset.
> 2. **Whitening** in PCA so each principal component has unit variance, preventing large components from dominating.
>
> On the base 1,500 images, we run **K-Means** with k=6 (5 known + 1 extra cluster). This yields about 99% accuracy if we match cluster IDs to labels. Then we store that model's cluster centers for future reference."*

Slide 5: Novelty Detection Approach (≈2 minutes)

> "Next, a **new batch** of 40 images arrives. We suspect exactly **one** new defect. Our pipeline is:
>
> 1. Preprocess & extract **fc1** → shape (40, 4096).
> 2. **PCA transform** from 4096→50 using the same PCA model as before.
> 3. **K-Means** with k=2 on those 50D features. We now get 2 new cluster centers.
> 4. Compare each new center's distance to the **old cluster center #0** (which we label 'Crazing'). Whichever new center is **closer** is labeled 'Crazing,' the other cluster => **'new class.'**
>
> This automatically flags half the new batch as something we've never seen. If we see more unknown types in the future, we might adapt or expand this approach."*

Slide 6: Live Demo & Results (≈2 minutes)

> "For a quick **demo**:
>
> 1. We load a new script that reads these 40 images.
> 2. We run `extract_fc1_batch(...)`, do `pca.transform(...)`, and cluster them with `kmeans_2`.
> 3. We see one cluster's center is ~3.5 distance from old center #0 → we label it 'Crazing'. The other cluster is ~7.1 away → 'new class'.
>
> **Figures**:
>
> - A **t-SNE** scatter of all 40 samples in 2D, color-coded as 'Crazing' vs. 'new class.' We

> - see two well-defined subgroups.
> - Optionally, we can embed a few sample images as insets to illustrate the difference visually."*

(You can quickly show code snippets or an actual Jupyter cell if time allows.)

Slide 7: Possible Improvements (≈1 minute)

> **"Finally, a few **improvements** we could make:
>
> 1. Instead of forcing exactly one 'Crazing' center, we might define a distance threshold that if both new centers exceed it, they're both new.
> 2. If multiple unknown classes appear simultaneously, we'd want a bigger k or a dynamic approach.
> 3. We might unify this with an **online** approach, so each new batch can update the K-Means model.
> 4. We can integrate a user confirmation loop: if the system detects a new cluster, a human can label it for future retraining."*

Ending & Questions

> **"That concludes our 10-minute overview. We introduced the problem of detecting 5 known defects + 1 new with a VGG16 + PCA + K-Means pipeline. We demonstrated the approach with the NEU dataset, a small script, and some t-SNE figures.
>
> **Thank you for listening.** I'm happy to take any questions or clarifications now."*