**Meeting Notes**

### Keyers and eCrash Performance

* The number of keyers is proportional to the performance of eCrash.
* If eCrash performance is good, fewer keyers are needed as they won't need to devote more time to manually check.

### Prioritization and Timeline

* Priority changes will affect the timeline. For example, focusing on code improvements can delay the plan to migrate to the new architecture, which prolongs addressing issues with the current architecture.
* A simpler timeline without technical details can be created, but detailed information is still requested.
* We need to determine the support duration and the timeline for tool improvement.

### Support vs. Performance Improvement

* We should ask which to prioritize: supporting the tool or improving performance?
* Doing immersion and development work with Strive can speed up knowledge transfer, benefiting tool support while using the current infrastructure.
* Migration and creating the new architecture are inevitable, even if we fully support or own the tool, as we need to address problems with the current architecture.
* A balance between immersion and migration can be struck. Immersion can serve as data gathering (e.g., 3 months of collaboration), then minimize tickets taken each sprint from Strive to have more bandwidth for new architecture development.

### Timeline and Model Improvement

* The current timeline does not consider immersion or development work with Strive while working on the new architecture, hence it was shorter.
* Model improvement was not prioritized in the current timeline as it was assumed Strive would handle this.
* Bala mentioned an overhaul in the model to address an issue; more details are needed.

### Management Objectives and Prioritization

* We need to ask upper management what their objectives are and what they want to prioritize: hiring the necessary number of keyers, improving the project, or ensuring we own and support the project as quickly as possible.
* If management asks us what to prioritize, it’s already answered in the current timeline: we want to improve the architecture to make it easier to work on errors and performance issues, improve workflow, reduce time spent addressing bugs or retraining the model, and gain visibility on pipeline issues with robust monitoring and logging.

### Collaboration with Strive

* We need to be proactive in communicating with Strive about collaborations to gauge their responsiveness and participation.
* The fact that Strive created a cloud migration plan indicates they know the current architecture isn't sufficient for future scaling and can't properly address issues (e.g., CI/CD, monitoring, logging, testing).
* It's a red flag that Strive initially deployed the project to production using JupyterLab for running scripts.
* We need to understand how Strive operates, suggest improvements, and have visibility and a voice in the project's direction.
* We need to know if Strive is inflating the time commitment for an issue.