

Final Project CM3070

PROJECT DEVELOPMENT & DEPLOYMENT ROADMAP

Given the simplified scope of your project, it's more feasible to complete it within 3-4 weeks. Here's a breakdown of the tasks and a suggested timeline:

Week 1: Project Setup and Motion Detection Integration

Goals: Set up the Django project, integrate existing OpenCV files for motion detection, and implement a basic UI.

Tasks:

1. Day 1-2: Project Setup

- Set up the Django project environment.
- Create a basic Django app with necessary configurations (database, static, and media files).

2. Day 3-4: Integrate OpenCV for Motion Detection

- Incorporate your existing OpenCV motion detection scripts into the Django project.
- Create a Django view to process and display webcam feeds with motion detection.

- Test the motion detection integration with your webcam.

3. Day 5-7: Build a Simple UI

- Design a basic frontend using Django templates.
- Create a page to display the live webcam feed with motion detection enabled.
- Set up the UI for viewing and managing recorded videos.

Deliverables:

- Basic Django project with OpenCV-based motion detection integrated.
- Simple UI for live feed and video management.

Week 2: Object Identification and File Management

Goals: Integrate TensorFlow for object identification, implement a file management system, and improve the motion detection capabilities.

Tasks:

4. Day 1-3: TensorFlow Object Identification Integration

- Set up TensorFlow and integrate it with your Django project.
- Update the motion detection script to include object identification.
- Test object detection and refine the model as needed.

5. Day 4-5: Implement File Management System

- Create a system for saving video recordings of detected motions.
- Set up Django models to manage and store video files.
- Implement views and templates to display and play recorded videos.

6. Day 6-7: Enhance Motion Detection

- Fine-tune your motion detection algorithm to improve accuracy, especially for webcam input.
- Add any additional features (e.g., sensitivity settings, area of interest) that improve detection.

Deliverables:

- Object detection integrated with motion detection.
- File management system for recording and playing back video clips.
- Enhanced motion detection capabilities.

Week 3: Alert Notifications and Final Touches

Goals: Implement a simple alert notification system, finalize the UI, and conduct testing.

Tasks:

7. Day 1-2: Implement Alert Notifications

- Set up a basic notification system to alert users when motion or specific objects are detected.

- Use Django's email backend or a simple message display on the UI.

8. Day 3-4: Finalize the UI and UX

- Refine the frontend design for a better user experience.
- Ensure all features are accessible and the UI is intuitive.

9. Day 5-6: Testing and Bug Fixes

- Conduct thorough testing of the entire system, including motion detection, object identification, video playback, and notifications.
- Fix any bugs or issues that arise during testing.

10. Day 7: Documentation and Final Adjustments

- Write documentation for your project, including setup instructions and user guides.
- Make any final adjustments or optimizations.

Deliverables:

- Simple alert notification system integrated.
- Finalized UI with improved UX.
- Fully tested and functional application.

Summary

This streamlined roadmap should allow you to complete the project within 3-4 weeks, focusing on key features without overcomplicating the scope. Consistent effort and focus on core functionalities will help you meet the deadline.

