Outlook

[CS 488T] Sprint 10 Report, Team 11 [stewartc]

From shelbyemailrelay@gmail.com <shelbyemailrelay@gmail.com>
Date Sat 4/12/2025 9:23 AM
To Stewart, Caleb <cstewart15@ewu.edu>

Caleb,

This report describes the activities of your EWU Senior Project team over the previous self-evaluation period (usually Saturday through Friday). It contains only public information. Private information and comments, etc. are available only to the instructor. If you notice any discrepancies or have questions, please contact Dan Tappan at dtappan@ewu.edu.

Sprint 10 Team Report

Team 11: Trademark ID & Analysis Engine

· Lane Keck
· Caleb Stewart
· Logan Taggart

Logged Hours

The team is generally free to work whenever they want during the sprint. The expectation for a team of three members is 45 hours total (15 per member) on average. However, this number will vary throughout the course.

Individual Hours:

		All Sprints							
Member	Hours	Total	Min	Max	Avg[1]	Avg[2]	Std[2]	Count[1]	Missed
Keck	8.0	67.0	3.0	9.0	7.4	7.4	1.7	9	0 (0%)
Stewart	9.0	78.0	2.0	16.0	8.7	8.7	3.8	9	0 (0%)
Taggart	8.0	63.0	3.0	10.0	7.0	7.0	2.4	9	0 (0%)
Team Total:	25.0								

[1] including and [2] excluding missed submissions for required sprints

Team Hours:

Sprint														
1	2	3	4	5	6	7	8	9	10	Total	Min	Max	Avg	Std
8.0	27.5	24.5	28.5	25.5	22.0	18.0	0.0	29.0	25.0	208.0	0.0	29.0	20.8	9.1

The following is optional descriptions of daily work that is not captured as activities below:

Taggart:

- Minor changes to codebase to improve clarity and modularity
- Working on MOSSE tracker to interpret logo position
- Fixing backend termination bug and learning about MOSSE tracker

Activities

Activities are member-defined units of work that are formally tracked from sprint to sprint (unlike the optional descriptions above). Every activity must be accounted for from its creation until it is completed or abandoned.

New Activities

These activities were created by during this sprint.

Keck

Activity 106: Connecting new front end to backend

Using the new video interface, make it functional (two sprints expected)

Stewart

Activity 101: FAISS General Search Working

Prove that FAISS can work. Write this code in an interactive notebook for testing purposes (one sprint expected)

Activity 102: Implement FAISS General Search

After FAISS works in interactive notebook, implement it into actual backedn (one sprint expected)

Activity 103: FAISS Specific Search Working

Get a working prototype of a specific logo search while using FAISS. for video search (two sprints expected)

Activity 104: Implement FAISS specific search

After FAISS works in interactive notebook, implement it into actual backedn (two sprints expected)

Taggart

Activity 105: MOSSE Tracker

Implementing MOSSE tracker to interpret logo boundary boxes in between analyzed frames. (two sprints expected)

Continuing Activities

These activities were continued from the previous sprint.

Activity 93.1: Style new stats that are returned

Opened in Sprint 7 by Keck; expected to take two sprints.

Original description: Make the stats look better

Progress in Sprint 9 (expected to take one more sprint): The stats are on the webpage but could still look better

Progress in Current Sprint (expected to take one more sprint): Still working on it

Activity 100.1: Work on video analysis

Opened in Sprint 9 by Keck; expected to take two sprints.

Original description: Get our model working with video

Progress in Current Sprint (expected to take one more sprint): We have the website set up for sending video, need to implement code in backend

Activity 97.1: Make video analysis more efficient

Opened in Sprint 9 by Stewart; expected to take two sprints.

Original description: Make processing videos less time consuming

Progress in Current Sprint (expected to take two more sprints): Using FAISS should reduce the runtime of processing a video

Activity 98.1: Video Processing

Opened in Sprint 9 by Taggart; expected to take three sprints.

Original description: Get YOLO model to work on videos

Progress in Current Sprint (expected to take two more sprints): The model is now working on videos and detects logos within frames, but still needs to be actually implemented into the backend code.

Activity 99.1: FAISS

Opened in Sprint 9 by Taggart; expected to take two sprints.

Original description: Learning and using FAISS to detect when logos in different frames are the same.

Progress in Current Sprint (expected to take one more sprint): FAISS now detects first instance of logo then stores it for similarity lookup later. This works in our testing but still needs to be actually implemented into backend code.

Completed Activities

These activities were completed during this sprint.

Activity 95.1: Intermediate presentation

Opened in Sprint 9 by Stewart; expected to take one sprint.

Original description: Collecting documents and construct presentation for intermediate presentation

Progress in Current Sprint: Presentation was successful

Activity 96.1: Start video analysis

Opened in Sprint 9 by Stewart; expected to take two sprints.

Original description: Get the model to work on a video

Progress in Current Sprint: We have a prototype of our YOLO model working on a video

Team Reflection

This section refers to the team's collective perception of and reflection on the project over this sprint.

The instructions are: Consider the following four pairs of questions hierarchically. They are not the same question. If you think they are, then you are likely not using an appropriate breadth and depth of software-engineering thought. This course is a practical application of the aspects of product, process, and people. We are trying to account for everything: not just to create a good product, but also to learn from the process to improve the people. Reflect on the experience of the entire team collectively over this sprint. You do not need to account for all work, just two examples that are most representative of easiest and hardest.

For reference, *understand* relates to the comprehension of what needs to be done; *approach* to how you think it should be solved; *solve* to implementing the actual solution; and *evaluate* to demonstrating to yourself and your team (if applicable) that the performance of your solution is consistent with everything else in the project. Remember The Cartoon from CS 350.

Understand

Easiest: It is easy to understand that we need to implement the code to make video analysis work. We also understand how to update the front end to match the video upload that we need

Hardest: It is hard to understand how to make the model run faster on videos. Currently, it runs slow which we might not be able to fix but this will take time to fully understand

Approach

Easiest: It is easiest to approach the frontend work that needs to be done because all we have to do is reuse the existing frontend components and modify them

Hardest: It is hardest to approach how to make the code more efficient. The model works just fine but somehow we need to process videos quicker.

Solve

Easiest: It is easiest to solve the implementation of our pseudocode into our backend. We already have existing code in a google colab notebook but now we must transfer this code over

Hardest: It will be hardest to solve the problem of returning our metrics to the user. We must figure out how many frames a logo appears in. We must also get the code working for 'Specific logo search'.

Evaluate

Easiest: It is easy to evaluate whether or not our code is functioning correctly. The frontend and backend will send error codes.

Hardest: It is hard to evaluate why the model is or is not detecting a logo. It is also hard to know why it takes so long to analyze a video as an image finishes in milliseconds

Completion: 70

Contact: N/A

Comments: No

Report generated on Sat Apr 12 09:23:31 PDT 2025