Project Development Plan
4ZP6: Die Spy

Team #9

Jackson Cassidy
Wyatt Habinski
Christian Majid
Paul Puscas

October 25, 2024

Revision History

Date	Version	Notes
October 25, 2024	0	

Team Meeting Plan

Our team has established a clear and efficient meeting structure to ensure the success of our capstone project. We will operate within an agile sprint methodology, utilizing Airtable to manage sprints and task assignments.

Air table allows us to write in our tasks and deadlines and converts it into the built in calendar function to visualize upcoming due dates. It provides a dashboard for our features, that we can assign to team members, and create tasks related to each feature. There is also a dashboard to track any issues we are facing, and link them to features, tasks, and team members respectively.

Our work will be organized into bi weekly sprints, each featuring task breakdowns, assigned responsibilities and set deadlines. This structure ensures that progress is continuous and measurable

We will maintain weekly discussions to handle day to day progress updates, quick check-ins, and resolving immediate questions. In addition, we will hold bi weekly sprint meetings to review our progress more formally. During these meetings we will:

- Discuss completed tasks and review the sprint progress
- Address any blockers or challenges the team is facing
- Assign new tasks and responsibilities for the upcoming sprint
- Update timeline and ensure that we hit our milestones

Team Communication Plan

Our team has multiple channels of communication to ensure efficient collaboration throughout the project. We utilize the following tools to keep everyone informed and aligned:

Discord

Our primary communication platform is Discord, where we maintain an open line of communication. We use it for daily discussions, updates, and problem solving. It provides an organized layout that allows us to share ideas and give informal updates through both chat rooms and voice channels.

Google Drive and Docs

For file sharing and collaborative document editing, we use Google Drive. All project related files including design documents, reports and dataset files are stored here, allowing team members to access and edit in real time. Google Docs ensures real time collaboration and a cohesive version history.

Airtable

We use Airtable for sprint planning and task management. It helps us organize and visualize our bi-weekly sprints, assign tasks, set deadlines and track progress.

Github

For code collaboration, we use Github. Our project code is shared through a repository, where team members can push, pull, review code, and manage version controls ensuring that code merges are smooth.

Team Member Roles

Coordinator/Program Manager: Christian

P	**Requirement**	**Responsible Team Member(s)**
P0	Camera functionality	Christian, Wyatt
	Dataset	Jackson, Paul, Christian, Wyatt
	Detect and Identify D6s, Count number of D6s, Sum D6	Paul, Jackson
	Solo Functionality	Christian, Jackson, Paul, Wyatt
P1	Scan from Live Video Feed	Christian, Paul
	Login screen, option to Host or Join	Jackson, Wyatt
	Connect to host and view results	Wyatt, Christian, Paul
	Maintain account/turn system, i.e track rolls by who’s turn it is	Jackson, Wyatt
	Editing dice roll (in case of mistake)	Christian, Paul
	Live Chat	Paul, Jackson, Christian, Wyatt

P2	Log dice roll results	Jackson, Wyatt
	Save previous game history	Wyatt, Jackson
	Upload to google play store	Christian
	D20, D12, D10, D8 Support	Paul, Jackson
	Statistics by account	Paul, Jackson, Christian, Wyatt
P3	Join the host from web	Christian, Wyatt, Jackson
	Highlighting dice in camera mode	Wyatt, Paul
	D4 Support	Paul, Jackson, Christian, Wyatt
	Send live footage to guests too	Paul, Jackson, Christian, Wyatt
	Simulate Roll option	Paul, Jackson, Christian, Wyatt

Workflow Plan

We will be using GitHub as our main software managing platform. We will be working on branches based on functionality or major milestones. We will always maintain a working or live branch that has our latest fully functioning prototype, so we can revert back to a fully working model. Pull requests will be reviewed by the team before any merges occur that way everyone is notified and aware of changes. Issues will be recorded on AirTable and marked into the documentation on GitHub. Any known issues at the time of launch will be transparent and accessible on the GitHub page.

As mentioned in the Team Meeting Plan we will use agile development methods, specifically bi-weekly sprints that we will use Airtable to organize.

We will utilize our personal computers to train AI models to perform heavy calculations. Between the four of us, we have more than enough power to train the model and create a working environment. During the development stage, data will be stored in the GitHub repository along with our code, in order to guarantee ease of access and a common version between all developers.

Our model evaluation will consist of a single testing script that can evaluate our agent, and measure the recall and accuracy of each agent. It will test our models on our test dataset, and a satisfactory performance by the models here will constitute having fulfilled the precision, accuracy, mAP, and F1-Score requirements.

Proof Of Concept Demonstration Plan

The primary risk to the success of our project lies in the accuracy and reliability of the machine learning models for detecting dice, identifying the upwards facing sides, and accurately reading the values from a video or image feed. These tasks must be performed in a variety of lighting and surface conditions, which could complicate the model's performance.

For the proof-of-concept demonstration, we will present an early version of our Android app that demonstrates the following:

1. Dice Detection: The app will use the device's camera to capture a static top down image of dice on a surface, and our machine learning model will identify and outline each die with bounding boxes. This will confirm the models ability to correctly identify multiple dice in the image.
2. Value Recognition: We will showcase the model identifying the value of each upward facing die, showing that the model can handle both numerical values and dots.

This demonstration will include a basic android app interface, which will display the results of the dice recognition and count. While the aim at this stage is limited to correctly identify D6's using a static top down image, we aim to prove that the remaining challenges, like live video feeds or multiple dice types, are within the realm of feasibility.

Technology

Programming Languages

Python - Used for machine learning models
Java - Used a bit for Android app development
Kotlin - Used as the main language for Android app development

For the front-end, we will be using mainly Kotlin with a little Java to create an Android application. For our back-end, which will mainly consist of the machine learning model, we will be utilizing PyTorch and various other libraries within Python. PyTorch is our machine learning library of choice due to its vast community support and documentation. PyTorch also allows us to train the model and then convert it to a LiteRT model, where it will be able to run efficiently on android. As previously discussed, Device-to-device communications will be handled by the Wi-Fi Direct standard, which is included as part of the Android Standard Library. Should any external network connections be required, they will be implemented using HTTPUrlConnection included in the android standard library. With regards to storing user data, due to the relatively low amount of data needed, we will likely store session and user data locally on the device of the host in a persistent form, such as writing to textual save files. We will likely use the JSON format for user data.

Unit Testing Framework

For android UI testing we will use Espresso. It has an easy to learn API, works with both Kotlin and Java, and is sufficient for the UI testing we expect to need. For non UI testing on android we will use JUnit since it has wide support, robust documentation, and many community resources. For our python code we will use pytest, for the same reasons as JUnit.

Versioning Standard

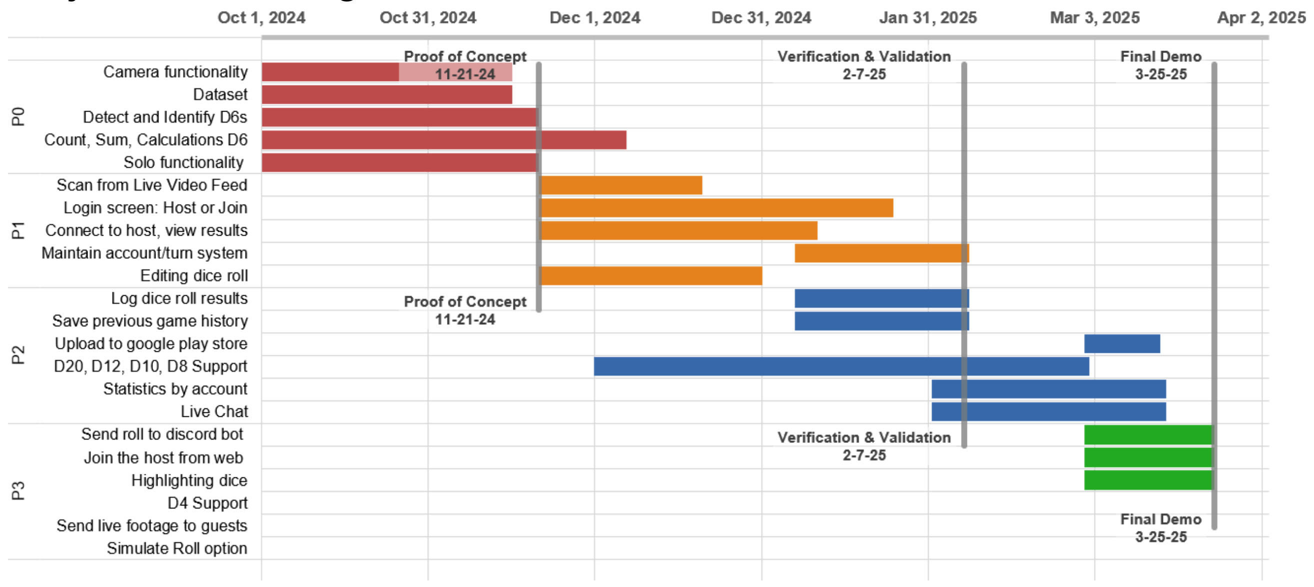
Python:	3.11.10
Pytorch:	2.5.1
Numpy:	1.26.4
Pandas:	2.2.3
Matplotlib:	3.9.2
Java JDK:	11.0.24
Kotlin:	2.0

GPU Use

For training and testing purposes, we have the capability of utilizing our own GPUs in our personal machine. This will support the GPU acceleration on our PyTorch model ensuring it runs at full capabilities. PyTorch is not dependent on a GPU, but using an Nvidia one will allow us to take advantage of its CUDA support. Utilizing our personal machines will also promote our own troubleshooting without external factors. As a backup we will use GPU runtime through Google Colab.

For live deployment we are going to run this locally on the device and the model should be compact enough to run on the phone. This can be accomplished by converting the trained PyTorch model to LiteRT, for example the LiteRT version of YOLOv5 is 7.61 MB.

Project Scheduling


Oct 1, 2024
Oct 31, 2024
Dec 1, 2024
Dec 31, 2024
Jan 31, 2025
Mar 3, 2025
Apr 2, 2025
P0
Camera functionality
Dataset
Detect and Identify D6s
Count, Sum, Calculations D6
Solo functionality
P1
Scan from Live Video Feed
Login screen: Host or Join
Connect to host, view results
Maintain account/turn system
Editing dice roll
P2
Log dice roll results
Save previous game history
Upload to google play store
D20, D12, D10, D8 Support
Statistics by account
Live Chat
P3
Send roll to discord bot
Join the host from web
Highlighting dice
D4 Support
Send live footage to guests
Simulate Roll option
Proof of Concept
Verification & Validation
Final Demo