Dwight Look College of
ENGINEERING
TEXAS A&M UNIVERSITY

ECEN 404 Bi-Weekly Update #4
Team 57: Deep Learning for Hydroponic Soybean Growth

Team members: Samuel He, Mary Hughes

Sponsor: Sambandh Dahl, Krishna Gadepally

Project Summary

- **Problem Statement:**

  Researchers take time to track the solution and day of growth of a hydroponically grown plant

- **Solution**

  Deep learning model and user interface that tracks Day of Growth and outputs other growth data that may be useful to the user

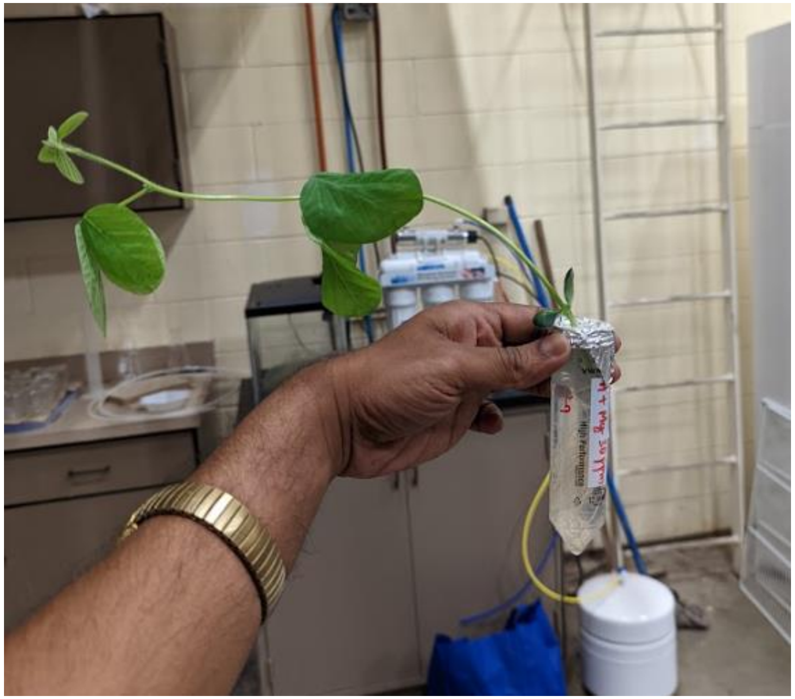
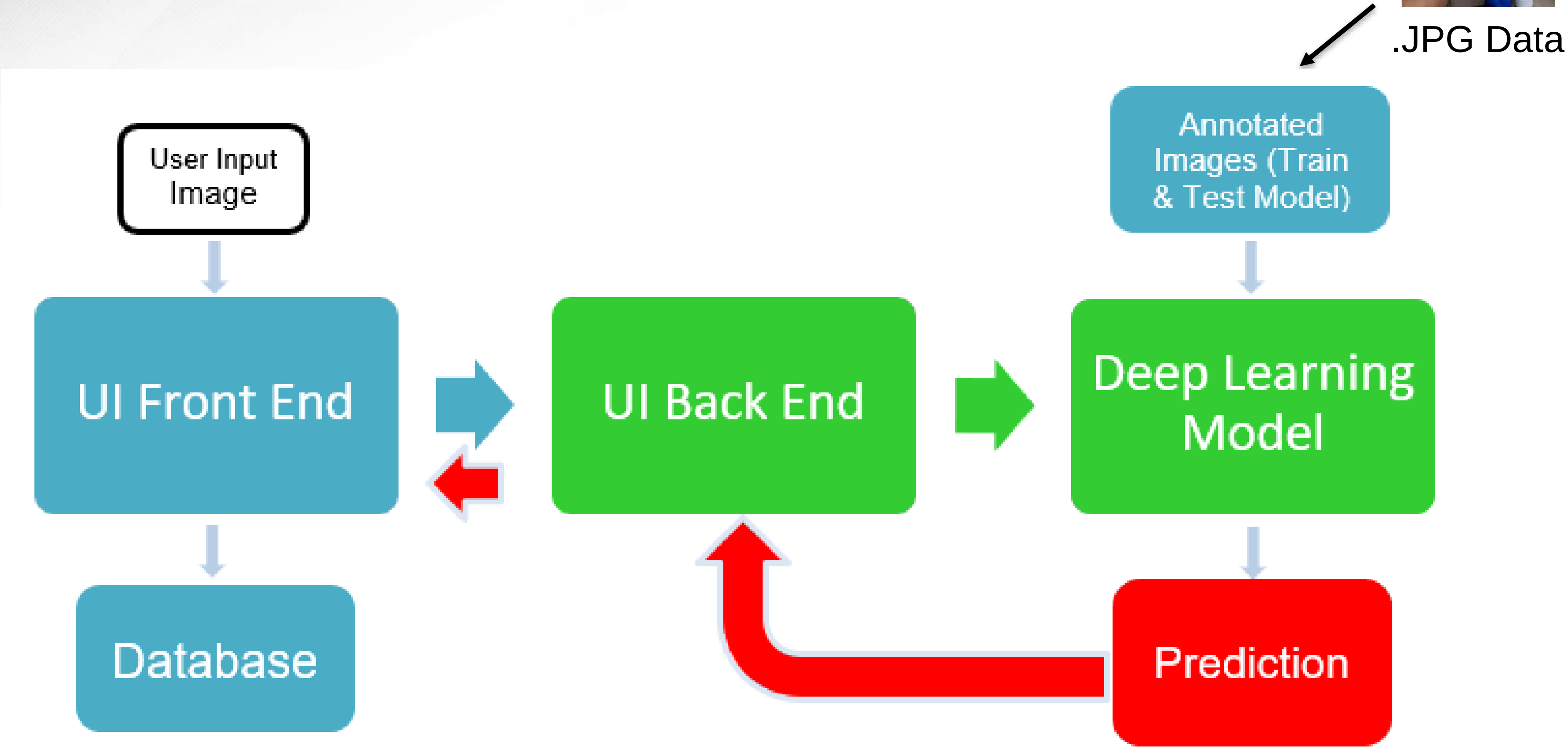


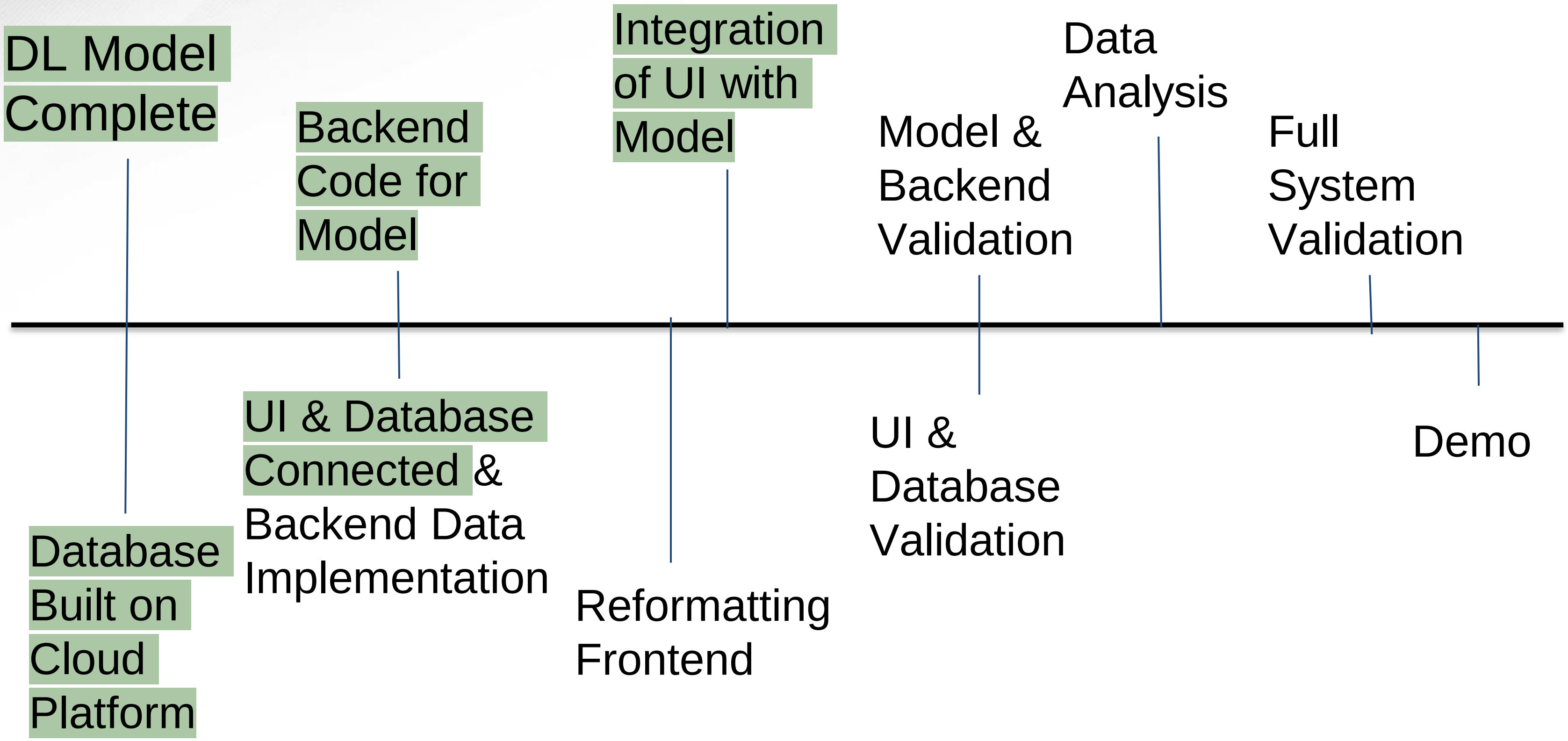
Image 1. Sample Data Image

System Diagram



Samuel He
Mary Hughes

Project Timeline



DL Model Complete

Database Built on Cloud Platform

Backend Code for Model

UI & Database Connected & Backend Data Implementation

Reformatting Frontend

Integration of UI with Model

Model & Backend Validation

UI & Database Validation

Data Analysis

Full System Validation

Demo

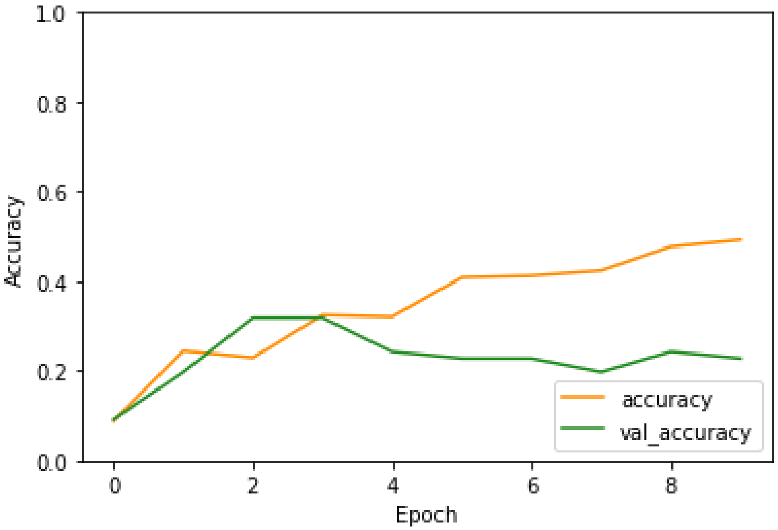
Subsystem: Deep Learning Model

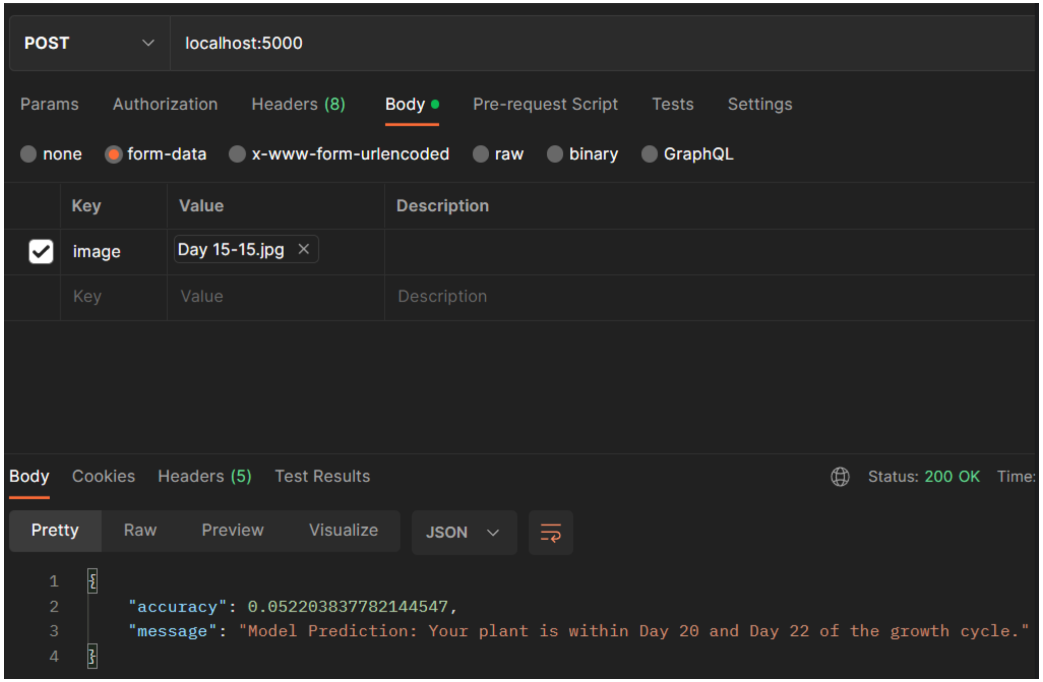
Accomplishments since 403 20 hours of effort	Ongoing progress/problems and plans until the next presentation
Integrated deep learning model and UI Implemented 3 Data Augmentation techniques within the model Changed output format to improve accuracy	Evaluate accuracy of the deep learning model Get deep learning model to make correct predictions Work with profs/TAs to get more techniques that will increase the prediction accuracy

Mary Hughes

Subsystem: DL Model

Training Accuracy & Validation Accuracy produced by training





POST localhost:5000

	Key	Value	Description
☑	image	Day 15-15.jpg	
	Key	Value	Description

Status: 200 OK Time:

```
{
    "accuracy": 0.052203837782144547,
    "message": "Model Prediction: Your plant is within Day 20 and Day 22 of the growth cycle."
}
```

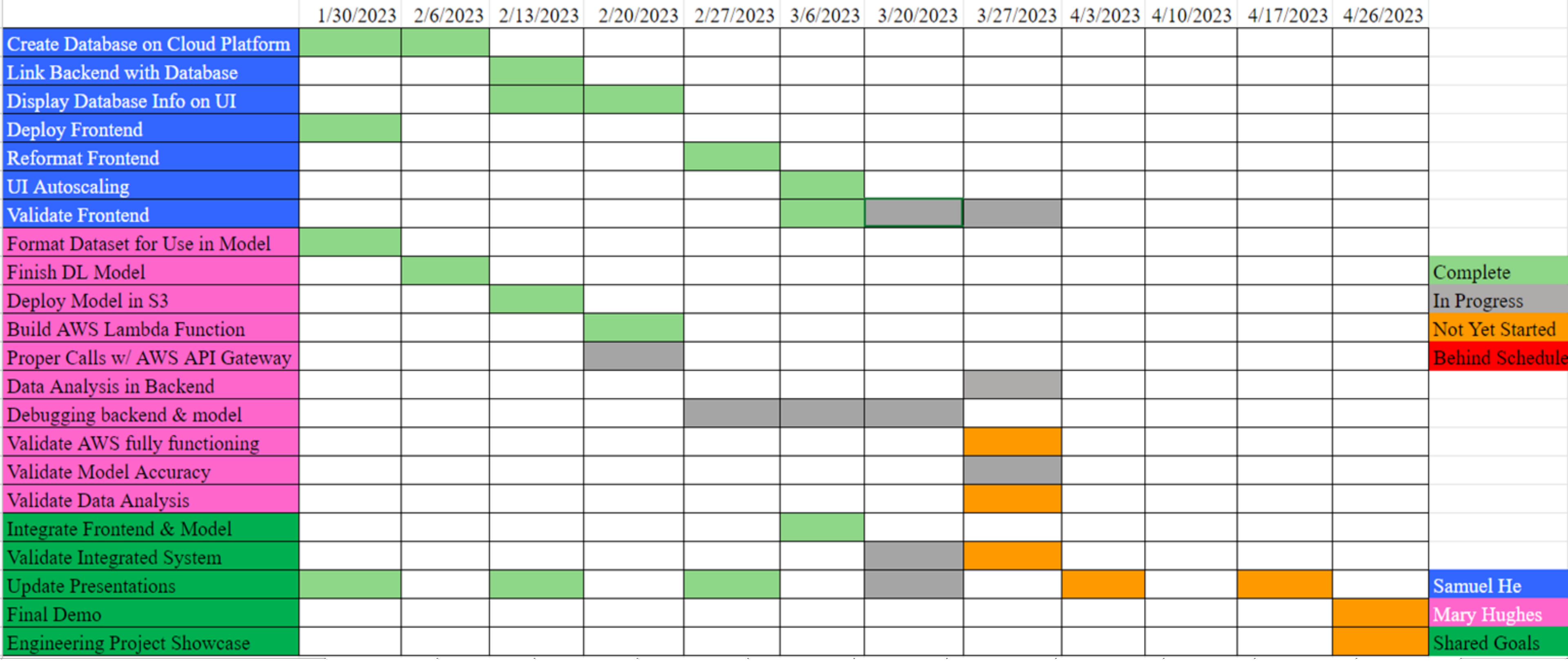
Local test using Postman: accuracy of the prediction is 5%, and the prediction is wrong.

Samuel He

Subsystem: User Interface

Accomplishments since last update 60 hours of effort (I have no life😢)	Ongoing progress/problems and plans until the next presentation
React: Finished all UI functionalities: - Input Form Structures for All 4 database tables - Uploaded Image Display Flask: Added an S3 Bucket - Uploads all files to an S3 bucket instead of directly in the database - Updated API to support new functions	Fix Heroku timeout issue with larger files Fix CSS Styling Grid layout issues Finish Validating UI - Inputting given Test data/Fixing Bugs

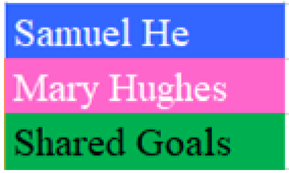
Execution Plan



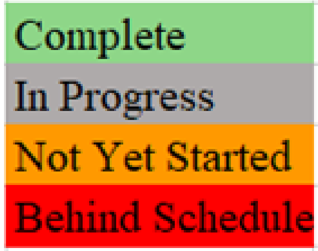
	1/30/2023	2/6/2023	2/13/2023	2/20/2023	2/27/2023	3/6/2023	3/20/2023	3/27/2023	4/3/2023	4/10/2023	4/17/2023	4/26/2023
🟦 Create Database on Cloud Platform	🟩	🟩										
🟦 Link Backend with Database			🟩									
🟦 Display Database Info on UI			🟩	🟩								
🟦 Deploy Frontend	🟩											
🟦 Reformat Frontend					🟩							
🟦 UI Autoscaling						🟩						
🟦 Validate Frontend						🟩	⬛	⬛				
🟪 Format Dataset for Use in Model	🟩											
🟪 Finish DL Model		🟩										
🟪 Deploy Model in S3			🟩									
🟪 Build AWS Lambda Function				🟩								
🟪 Proper Calls w/ AWS API Gateway				⬛								
🟪 Data Analysis in Backend								⬛				
🟪 Debugging backend & model					⬛	⬛	⬛					
🟪 Validate AWS fully functioning								🟧				
🟪 Validate Model Accuracy								⬛				
🟪 Validate Data Analysis								🟧				
🟢 Integrate Frontend & Model						🟩						
🟢 Validate Integrated System							⬛	🟧				
🟢 Update Presentations	🟩		🟩		🟩		⬛		🟧		🟧	
🟢 Final Demo												🟧
🟢 Engineering Project Showcase												🟧

🟩 Complete
⬛ In Progress
🟧 Not Yet Started
🟥 Behind Schedule

🟦 Samuel He
🟪 Mary Hughes
🟢 Shared Goals

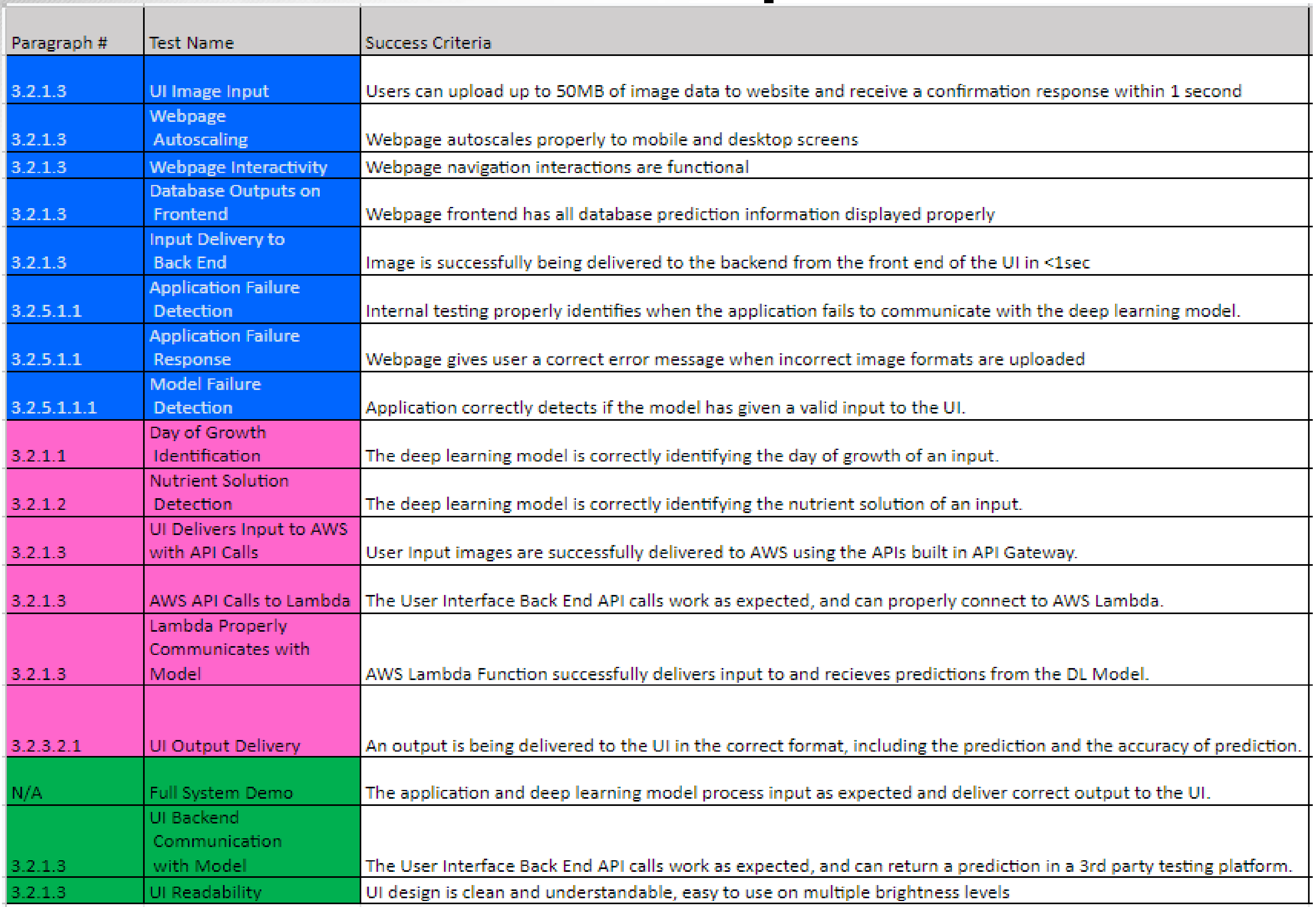


🟦 Samuel He
🟪 Mary Hughes
🟢 Shared Goals



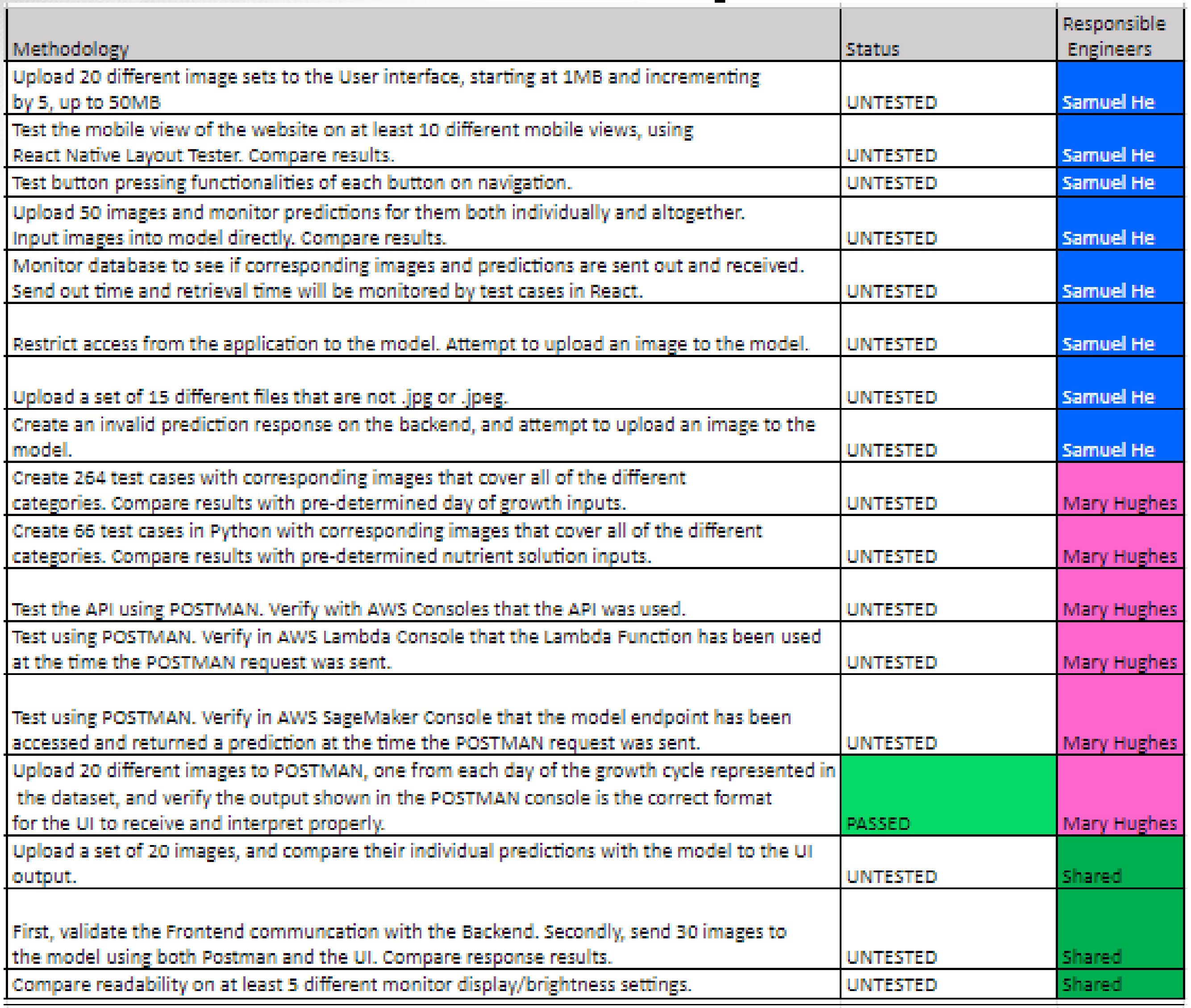
🟩 Complete
⬛ In Progress
🟧 Not Yet Started
🟥 Behind Schedule

Validation plan



Paragraph #	Test Name	Success Criteria
3.2.1.3	UI Image Input	Users can upload up to 50MB of image data to website and receive a confirmation response within 1 second
3.2.1.3	Webpage Autoscaling	Webpage autoscales properly to mobile and desktop screens
3.2.1.3	Webpage Interactivity	Webpage navigation interactions are functional
3.2.1.3	Database Outputs on Frontend	Webpage frontend has all database prediction information displayed properly
3.2.1.3	Input Delivery to Back End	Image is successfully being delivered to the backend from the front end of the UI in <1sec
3.2.5.1.1	Application Failure Detection	Internal testing properly identifies when the application fails to communicate with the deep learning model.
3.2.5.1.1	Application Failure Response	Webpage gives user a correct error message when incorrect image formats are uploaded
3.2.5.1.1.1	Model Failure Detection	Application correctly detects if the model has given a valid input to the UI.
3.2.1.1	Day of Growth Identification	The deep learning model is correctly identifying the day of growth of an input.
3.2.1.2	Nutrient Solution Detection	The deep learning model is correctly identifying the nutrient solution of an input.
3.2.1.3	UI Delivers Input to AWS with API Calls	User Input images are successfully delivered to AWS using the APIs built in API Gateway.
3.2.1.3	AWS API Calls to Lambda	The User Interface Back End API calls work as expected, and can properly connect to AWS Lambda.
3.2.1.3	Lambda Properly Communicates with Model	AWS Lambda Function successfully delivers input to and recieves predictions from the DL Model.
3.2.3.2.1	UI Output Delivery	An output is being delivered to the UI in the correct format, including the prediction and the accuracy of prediction.
N/A	Full System Demo	The application and deep learning model process input as expected and deliver correct output to the UI.
3.2.1.3	UI Backend Communication with Model	The User Interface Back End API calls work as expected, and can return a prediction in a 3rd party testing platform.
3.2.1.3	UI Readability	UI design is clean and understandable, easy to use on multiple brightness levels

Validation plan



Methodology	Status	Responsible Engineers
Upload 20 different image sets to the User interface, starting at 1MB and incrementing by 5, up to 50MB	UNTESTED	Samuel He
Test the mobile view of the website on at least 10 different mobile views, using React Native Layout Tester. Compare results.	UNTESTED	Samuel He
Test button pressing functionalities of each button on navigation.	UNTESTED	Samuel He
Upload 50 images and monitor predictions for them both individually and altogether. Input images into model directly. Compare results.	UNTESTED	Samuel He
Monitor database to see if corresponding images and predictions are sent out and received. Send out time and retrieval time will be monitored by test cases in React.	UNTESTED	Samuel He
Restrict access from the application to the model. Attempt to upload an image to the model.	UNTESTED	Samuel He
Upload a set of 15 different files that are not .jpg or .jpeg.	UNTESTED	Samuel He
Create an invalid prediction response on the backend, and attempt to upload an image to the model.	UNTESTED	Samuel He
Create 264 test cases with corresponding images that cover all of the different categories. Compare results with pre-determined day of growth inputs.	UNTESTED	Mary Hughes
Create 66 test cases in Python with corresponding images that cover all of the different categories. Compare results with pre-determined nutrient solution inputs.	UNTESTED	Mary Hughes
Test the API using POSTMAN. Verify with AWS Consoles that the API was used.	UNTESTED	Mary Hughes
Test using POSTMAN. Verify in AWS Lambda Console that the Lambda Function has been used at the time the POSTMAN request was sent.	UNTESTED	Mary Hughes
Test using POSTMAN. Verify in AWS SageMaker Console that the model endpoint has been accessed and returned a prediction at the time the POSTMAN request was sent.	UNTESTED	Mary Hughes
Upload 20 different images to POSTMAN, one from each day of the growth cycle represented in the dataset, and verify the output shown in the POSTMAN console is the correct format for the UI to receive and interpret properly.	PASSED	Mary Hughes
Upload a set of 20 images, and compare their individual predictions with the model to the UI output.	UNTESTED	Shared
First, validate the Frontend communcation with the Backend. Secondly, send 30 images to the model using both Postman and the UI. Compare response results.	UNTESTED	Shared
Compare readability on at least 5 different monitor display/brightness settings.	UNTESTED	Shared

Thanks & Gig ‘Em!