Wireless Inexpensive Bathymetry Logger (WIBL)

[1, 4] Patrick Doherty, [1, 3] Connor Murphy, [1, 4] Jason Worden, [1, 3] Jason Waleryszak, [2] Dr. Brian Calder

[1] Computer Science Department, University of New Hampshire, Durham, NH 03824
[2] Jere A. Chase Ocean Engineering Laboratory, University of New Hampshire, Durham, NH 03824
[3] Data Processing Pipeline
[4] Cross-Platform Mobile Application

Introduction

WIBL is one of CCOM JHC's contributions to the Seabed 2030 project, an international effort to produce a definitive map of the world's ocean floor by the year 2030. WIBL aims to provide an inexpensive system to contribute to this effort. The project consists of three primary components: the physical logger, used for collecting the bathymetry data; a mobile application, used for transferring the data from the logger to the data processing pipeline; and finally the data processing pipeline, used to refine, convert, and upload the data to DCDB, the data repository for Seabed 2030.

The objectives of this project were twofold: develop a cross-platform mobile application for transferring data from the logger to the pipeline, and replace the linear data processing pipeline with a modular alternative, enabling customization by third-parties.

Data Processing Pipeline

WIBL does not perform any on-device data refinements. Instead, refinement operations such as timestamping, deduplication, and outlier rejection are performed by a data processing pipeline in AWS. Figure 1 illustrates all components of the WIBL system, of note is the pipeline (encircled by the rectangle). Figure 2 presents the dynamic, non-linear pipeline which now replaces the pipeline included in Figure 1. The pipeline makes use of AWS Step Function, a service for chaining AWS Lambdas together using a state machine routing model. All data is stored within a singular WIBL bucket (not shown in the diagram). When data is uploaded to the bucket, AWS EventBridge triggers the step function, thus beginning a new pipeline execution. Jobs implement pipeline or data refinement operations. The controller handles which jobs are run, and when. The controller implements a routing strategy, allowing for both (simulated) linear and non-linear routing. To make all of this easy to deploy, Serverless has been configured to enable single-command deployment and setup of the entire pipeline.

Cross-Platform Mobile App

Loggers are not capable of uploading to the data processing pipeline directly, instead they utilize a mobile device to offload the data and upload it after a voyage has been completed. The app has to be able to connect to the physical logger, retrieve the bathymetric data stored on the logger, and upload that data to the data processing pipeline. WIBL switch from using Bluetooth to WIFI for data transfers just prior to the start of this project; this change resulting in the depreciation of previous app, which only supported Android. Google's Flutter was selected as the cross-platform framework for the app because of its speed and extensive documentation. The app fully supports this new mode of data transfer and provides support for all major mobile platforms.

Data Processing Pipeline Architecture

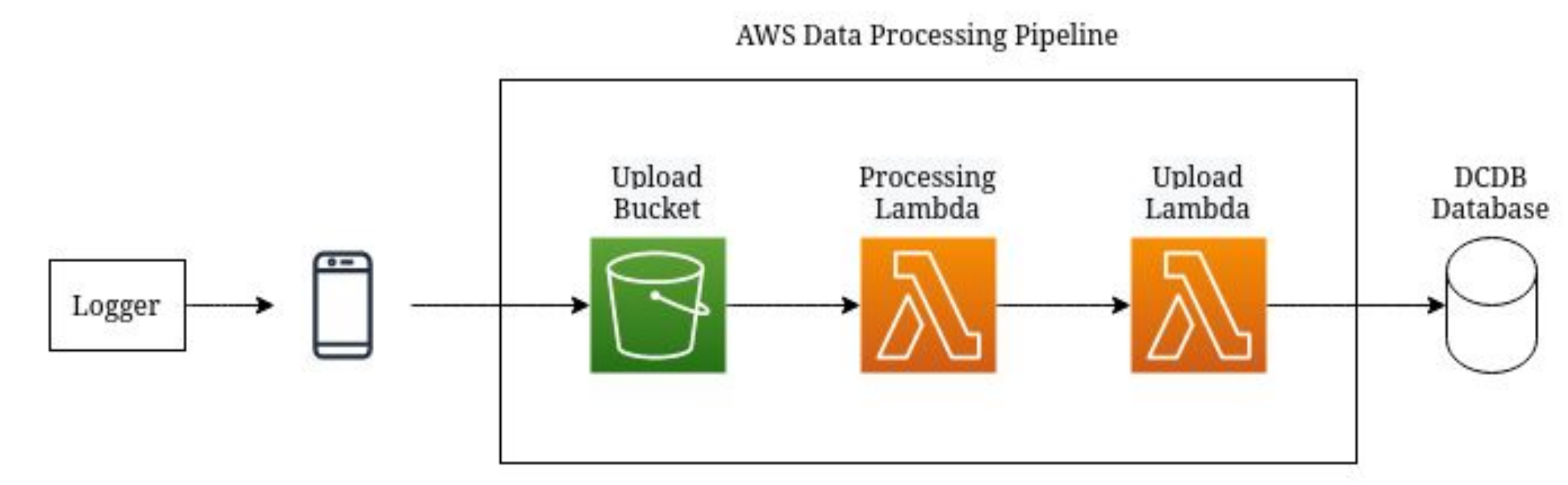

Figure 1: Upload process diagram implementing previous, non-configurable, AWS processing pipeline

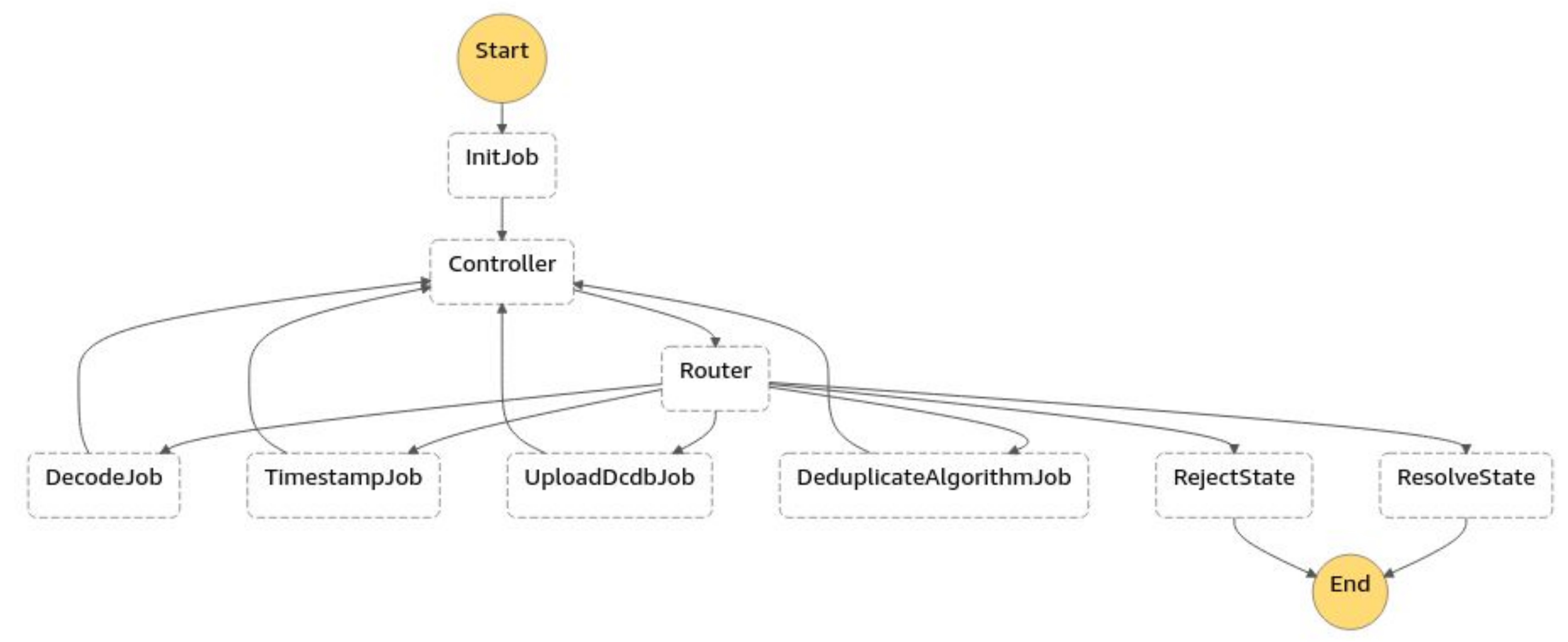

Figure 2: New AWS data processing pipeline using an AWS Step Function for routing and AWS Lambdas for compute operations

App Design

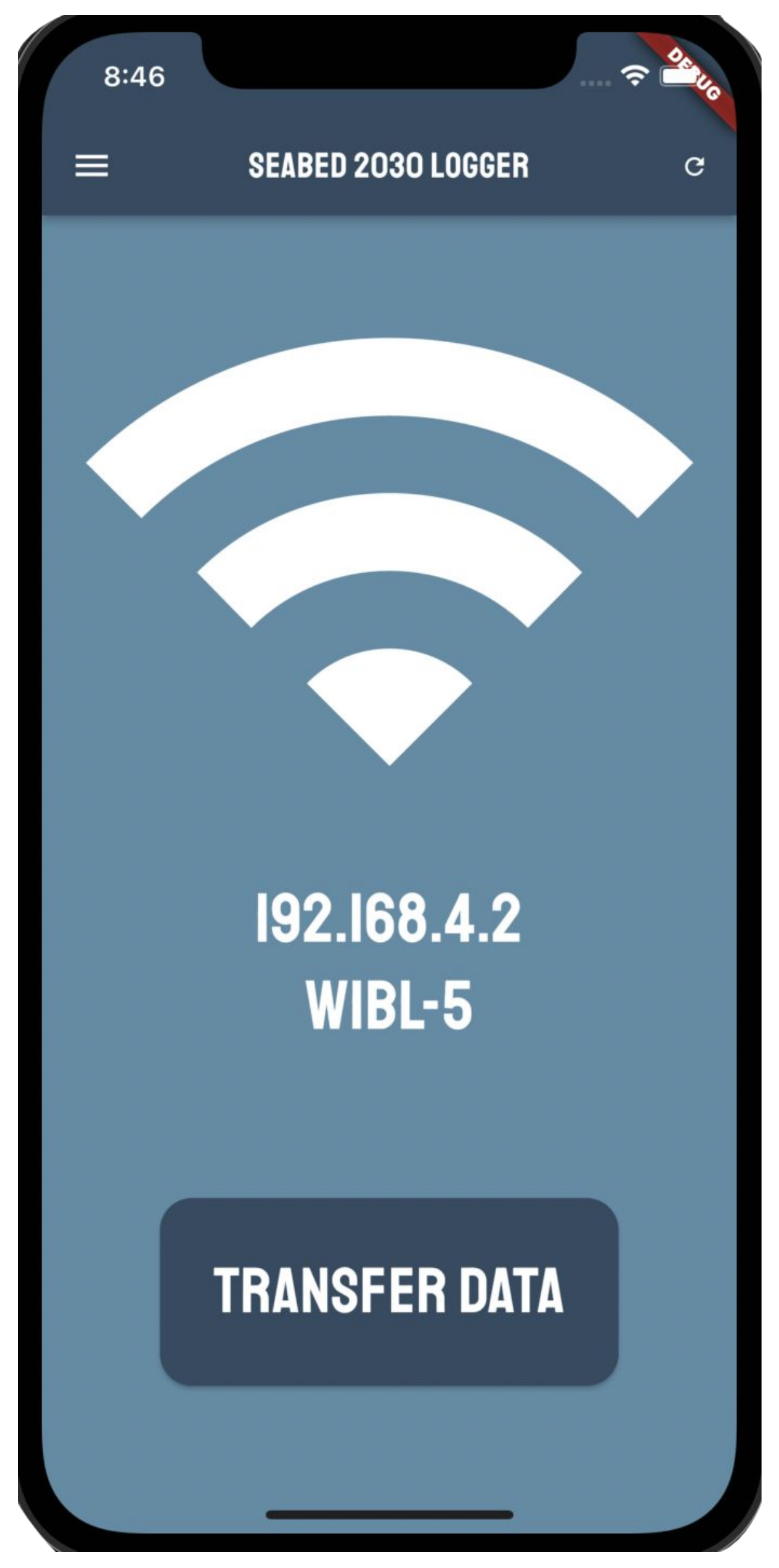

Figure 3: Cross-platform app data transfer screen

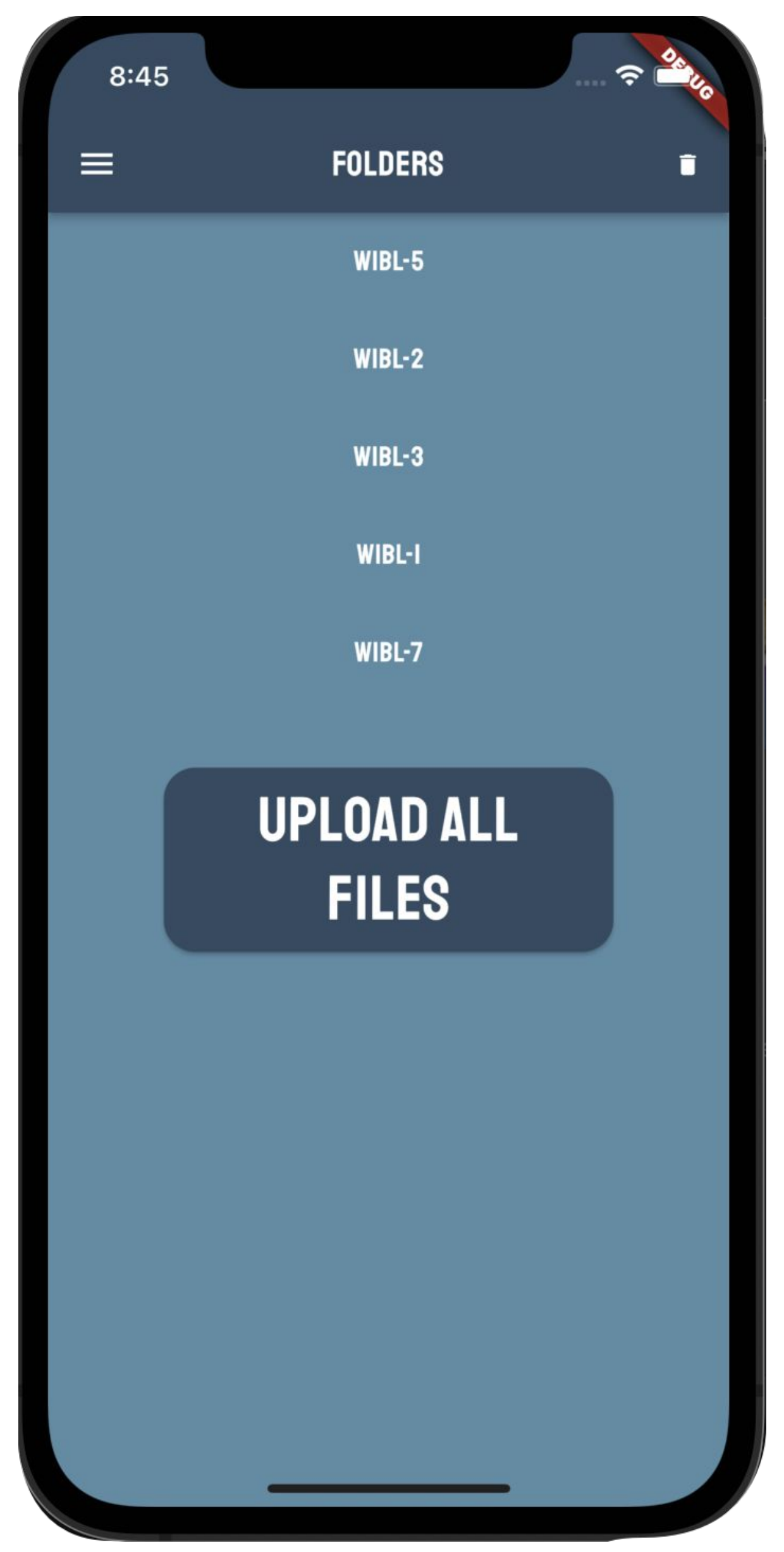

Figure 4: Cross-platform app configuration screen

The user first connects the mobile device to the logger via wifi. When the app detects a connection to a wibl device it displays the devices IP on the home page and provides access to the transfer button as shown in Figure 3.

When the "Transfer Data" button is pressed the app sends a command to the logger over wifi that returns all the files, which are then stored locally on the device and are visible in the Files tab.

When the "Upload Data" button is pressed in the Files tab all the files stored locally are transferred to the upload bucket in the cloud processing stack.

Figure 4 shows the Settings page used to configure the app. Each setting has a brief description underneath.

Status

Completed Features

- **Cross-platform Mobile Application**
  - Download data from the logger using IOS or Android device
  - Stage data on-device for future uploading
  - Delete stored logger data
  - Upload data from device to AWS bucket
- **Data Processing Pipeline**
  - Add support for data-driven processing routing
  - Implement framework for easy operation creation and customization of pipeline
  - Add single-step deployment of pipeline to AWS
  - Centralize instance-specific configuration

Future Work

- **Cross-platform Mobile Application**
  - Allow for deletion of specific files
  - Implement error catching when transferring
- **Data Processing Pipeline**
  - Create web-app for easily customizing configuration and deployment configs
  - Expand pre-existing documentation regarding custom operation creation and customization

Conclusions

The project achieved both of the goals outlined in the project's charter. The cross-platform app and new pipeline are both production ready and published as open source projects:

Pipeline WIBL App

WIBL will continue active development by Dr. Brian Calder and CCOM through the generous funding of NOAA.

Acknowledgements

Dr. Brian Calder, CCOM, Project sponsor
Matthew Plumlee, UNH CS Department, Advising faculty

Dr. Brian Miles, CCOM, Data serialization
Chris Schwartz, CCOM, Developer of WIBL Android app
Adriano Fonseca, CCOM, Logger setup and configuration
Joshua Girgis, CCOM, Logger setup and configuration