MICHIGAN STATE UNIVERSITY

Project Plan

Security Analytics Suite: Dataset Merger Tool

The Capstone Experience

Team Avata

Jonny Dowdall
Paige Henderson
Matt Scheffler
Aasir Walajahi
Zac Wellmer

Department of Computer Science and Engineering
Michigan State University

Fall 2016

From Students…
…to Professionals

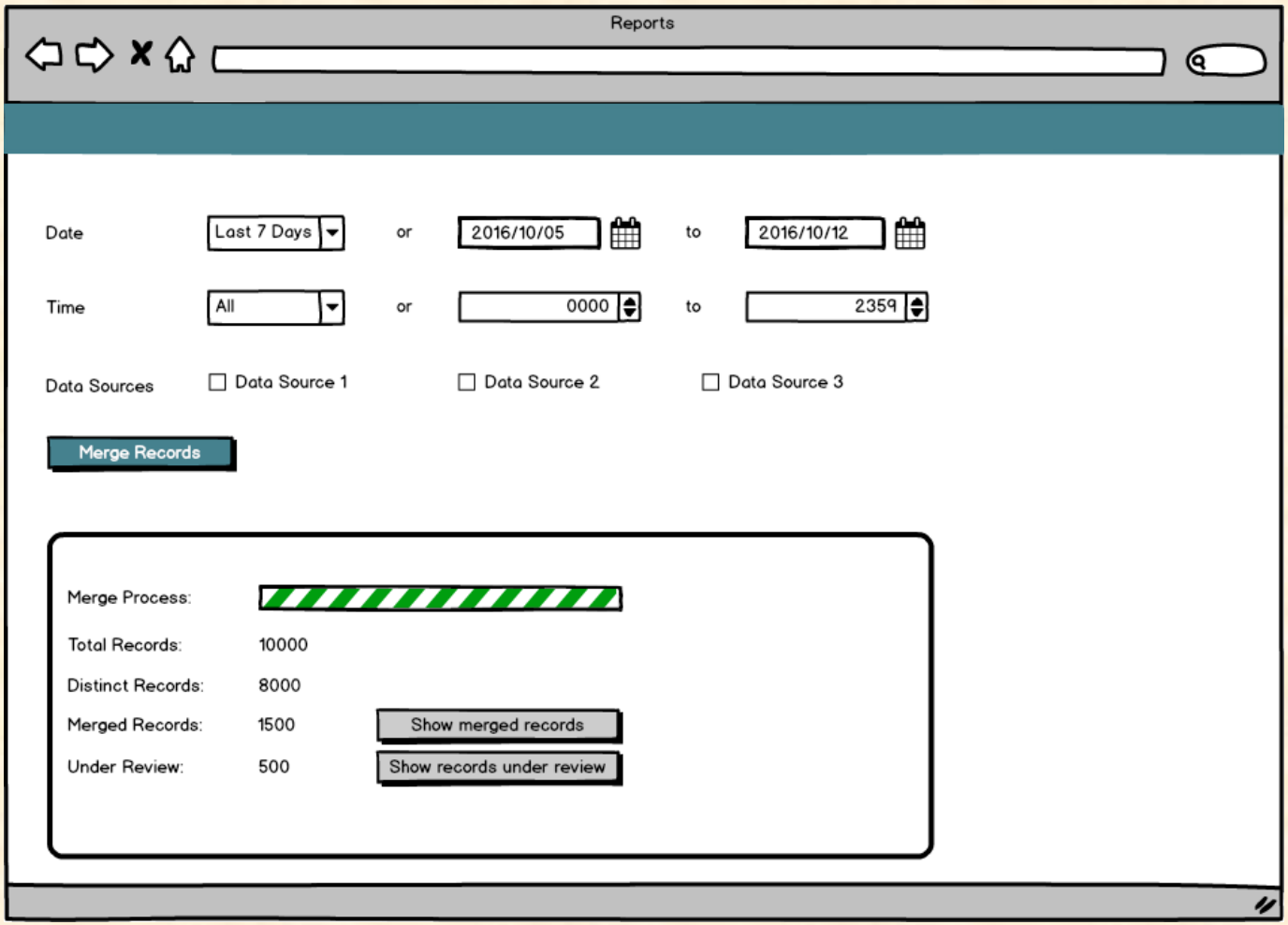
Functional Specifications

- Automatically identify and merge duplicate records within and across datasets
- Suggest similar records as potential duplicates to the user through responsive web application
- Allow the user to approve or disapprove the merging of similar records
- Present the results as a report containing tables and charts

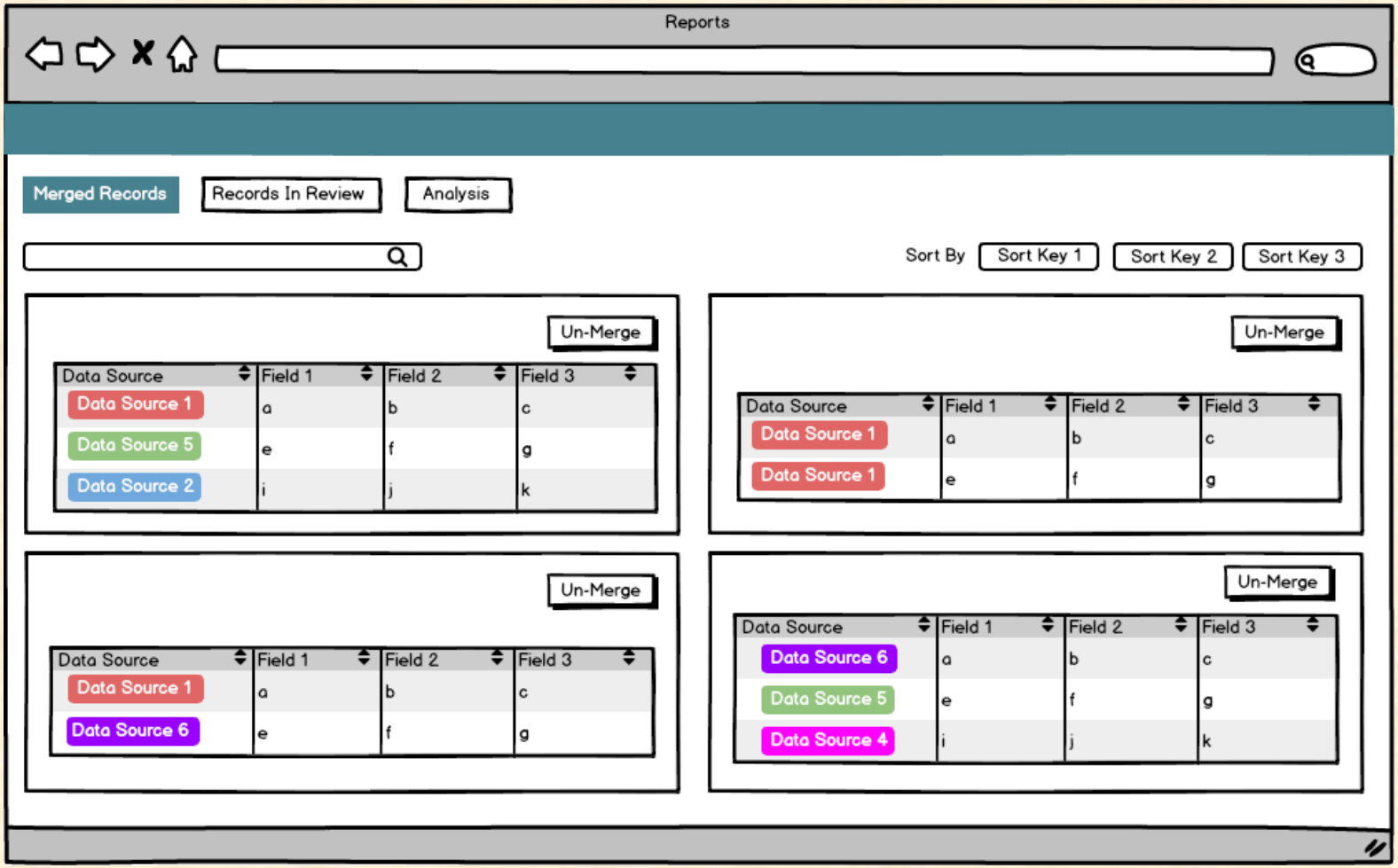
Design Specifications

- Run Page
  - Parameterized search (search within date range, select data sets, etc.)
  - Initiate merge job
  - Displays progress of running job
- Merged Records Page
  - History of merged records
  - Allows un-merging
- Review Page
  - Allows manual merging of suggested similar records
- Analysis Page
  - Displays merge process statistics

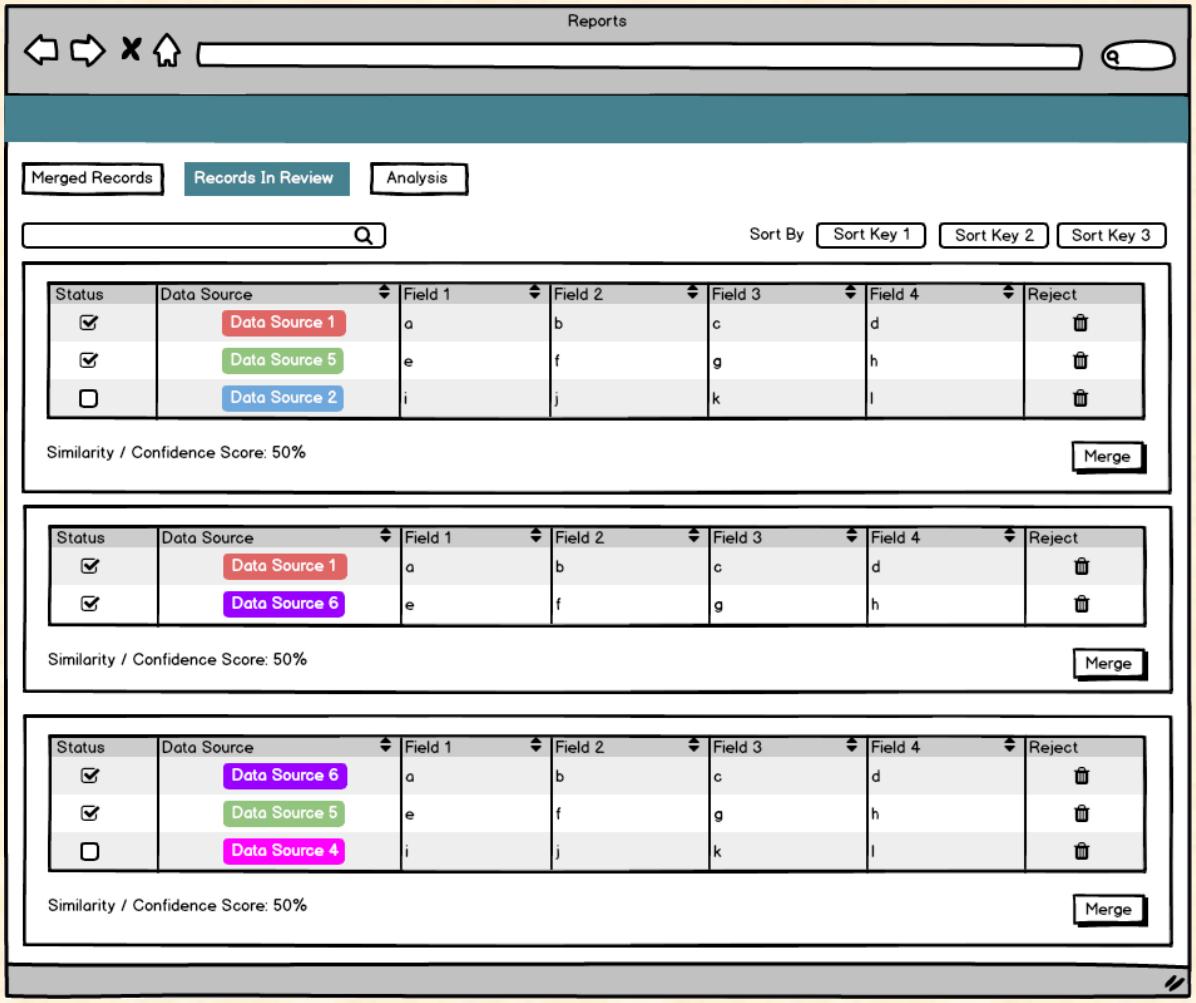
Screen Mockup: Run



Screen Mockup: Merged Records



Screen Mockup: Review



Screen Mockup: Analysis

Technical Specifications

- Front-end
  - Receives data from back-end to be displayed in view
  - Passes decisions about questionable mergers to the back-end
  - Orders controller to run the merge engine
  - Instructs controller to revert merged records from merge history table to original database

Technical Specifications

- Back-end
  - Written in Java and uses Spring Boot framework to connect to front-end and database
  - Utilizes model-service-controller architecture
  - Model: data structures
  - Service: Merge Engine
  - Controller: REST API

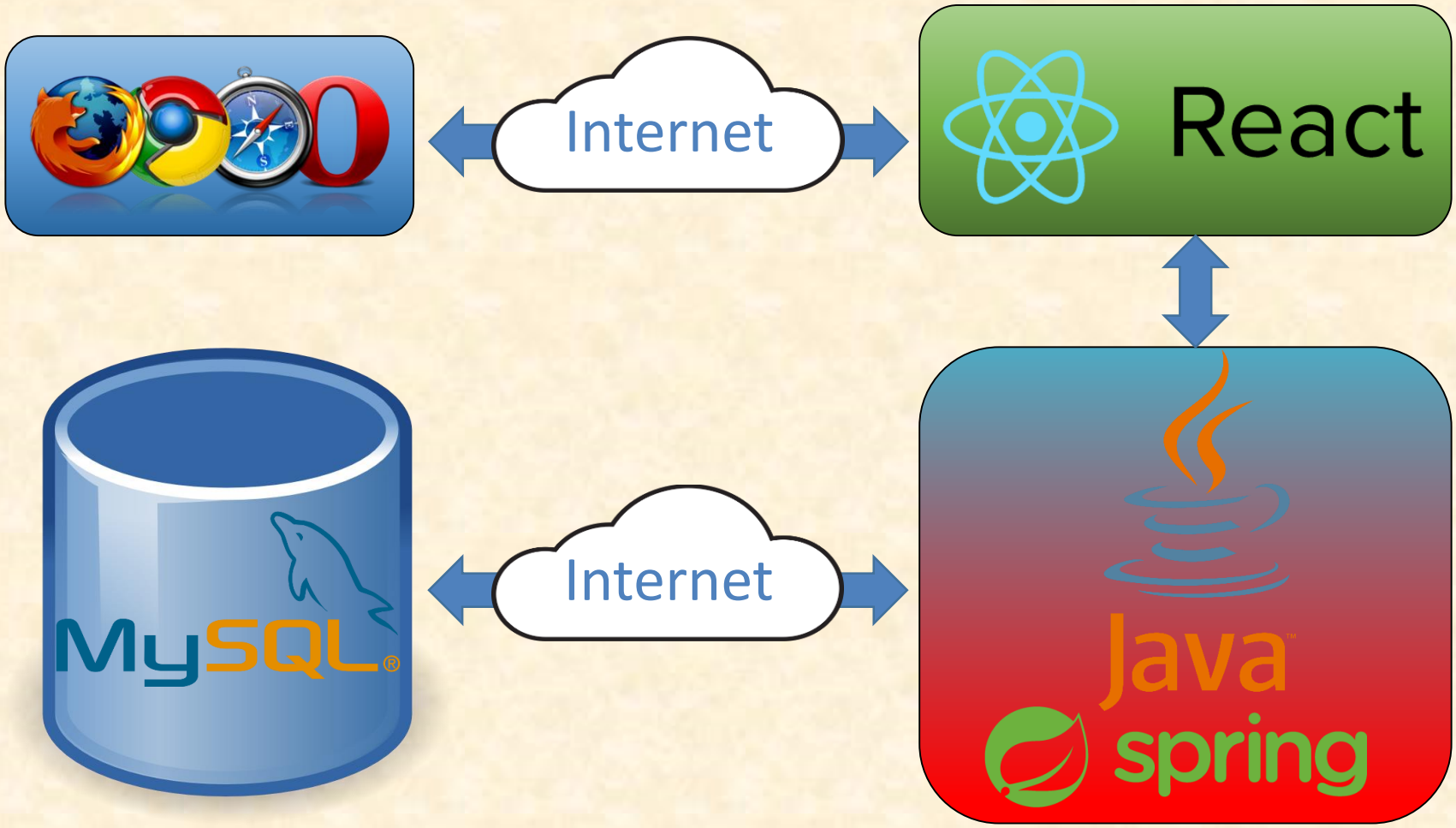
Technical Specifications

- Engine
  - Written in Java
  - Clears merge history table on run
  - Analyzes data with string-matching algorithm
  - Measure similarity between records and quantifies a “match score.”
  - Creates new record to represent two merged records and inserts into database
  - Moves two original records to merge history table

Technical Specifications

- Database & Server
  - MySQL Database
  - Hosted on Capstone server (Ubuntu 16.04)
  - Hosts original data and merge history table

System Architecture



System Components

- Hardware Platforms
  - Capstone server rack, Ubuntu 16.04 Server
  - iMac OSX
  - Windows Virtual Machine
- Software Platforms / Technologies
  - ReactJS - Javascript library
  - Flux - ReactJS application framework
  - Spring Boot - back-end Java framework
  - MySQL - database

Testing: Front-End

- User Interface testing
  - Application will be given to variety of new users
  - Users will test UI functionality through normal usage of app
- ReactJS Unit Testing
  - Unit testing will be performed using the Mocha Javascript Test framework
  - Unit testing on all major components
  - Records display tables, merge/un-merge buttons etc...

Testing: Back-End

- Unit Testing
  - JUnit - standard for unit testing Java applications
- Integration Testing
  - Spring Boot Test - utilities and integration test support for Spring Boot applications
  - Completed after all system components have been unit tested

Testing: Merge Engine

- Performance Evaluation
  - Error will be measured by evaluating the area under the Receiver Operating Characteristic (ROC) curve
  - ROC curve plots true positive rate against the false positive rate
  - Measure the probability a random positive sample will be scored higher than a random negative sample
  - Allow us to compare models
    - Models can vary on hyper-parameters such as confidence threshold

Risks

- Familiarity with new technologies
  - No one on the team has any past experience with ReactJS or Spring Boot
  - Mitigation: Researching and going through tutorials
- Catching duplicates with mismatched fields
  - We are not sure how we are going to match duplicate records with mismatched data (i.e. different formatting, mismatched fields for same records, etc.)
  - Mitigation: Research relevant algorithms and explore search capabilities of tools such as ElasticSearch

Risks

- Handling un-merging due to complexity of the data
  - We do not currently know the structure of the data provided by Avata
  - We may have difficulty devising a way to undo merges depending on the complexity of the data
  - Mitigation: Make sure our database is normalized and also use the client contacts as resources, as they have had experience working with these datasets