

IoT Remote Monitoring Application

For Washing Machines and Dryers

Team 17

SDDEC18-17



Team Members & Contributions

John Fleiner & Ben Young

- Android & iOS

Thomas Stackhouse & Casey Gehling

- Spring Boot Web Server
- AWS IoT

Hongyi Bian & Yuanbo Zheng

- Hardware Implementation

Advisor:

Goce Trajcevski

Client:

Greiner Jennings Holdings



Outline

1. Overview
2. Requirements
3. Considerations
4. System Design
5. System Implementation
6. Testing
7. Prototype

1

OVERVIEW



Project Overview

Problem Description

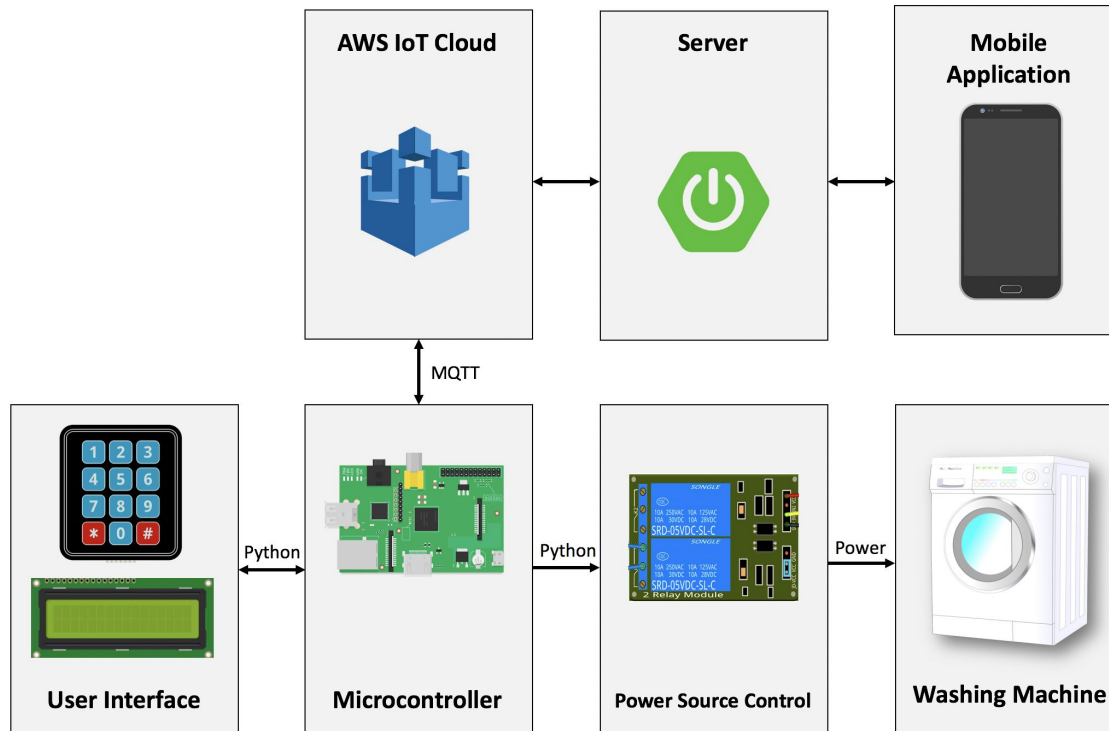
- Laundromats:
 - Managing Multiple Locations
 - Effective Scheduling

Project Goals

- Mobile Reservation Application
- Appliance Assignment Mechanism



Concept Diagram



2

REQUIREMENTS



Requirements Specification

Functional Requirements

- Create & Pay for Reservation
- Reservation Code → Unlock Appliance
- Reservation Ends → Lock Appliance



Requirements Specification

Non-Functional Requirements

- Secure Personal Data
- Authorized API Access
- Commercially Scalable

3

Considerations



Market Survey

IoT Pay-per-wash Industry

- Subscription based services

Microsoft Azure Berendsen IoT Hotel Laundry Service

- IoT tracking on linen ID tags

Samsung Electronic Laundry Innovation

- IoT compatible with SmartThing ecosystem



Resource Requirements

Items	Price
Amazon Web Service IoT	\$73.67
Portable Washing Machine	\$109.97
Raspberry Pi 3 Model B	\$35.00
16x2 LCD Display	\$9.95
3x4 Matrix Keypad	\$7.50
4-Channel Relay Module	\$6.97
Basic Electronic Kit	\$15.00
Total:	\$258.06



Risks → Mitigations

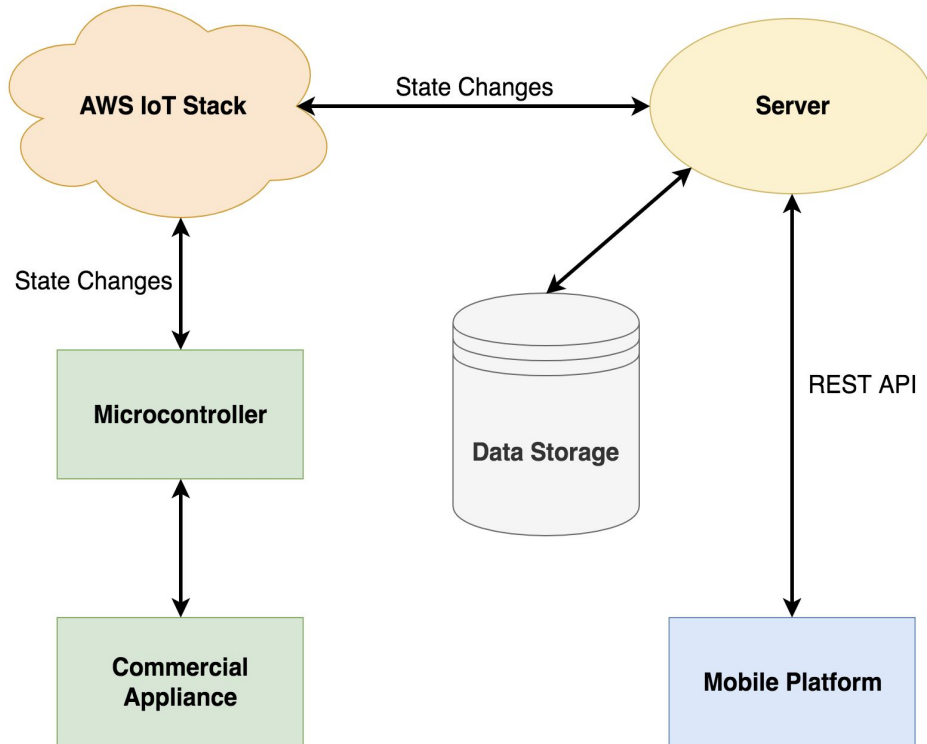
- **Privacy**
 - ▷ HTTPS, Spring Security Auth
- **Storing Credit Cards**
 - ▷ Database/Transfer Encryption
- **Accidental Soldering Errors**
 - ▷ Backup Hardware

4

System Design



System Architecture & Analysis



Mobile

- Android/iOS Native Apps

Backend

- Spring Boot Web Server
- AWS IoT
- MYSQL DB

Hardware

- Raspberry Pi 3
- Washing Machine
- Keypad & LCD



Design Trade Offs

■ Spring Server vs Calling DB Directly

- ▷ Centralized & Scalable
- ▷ Prior Experience
- ▷ Separating Visualization and Computation

■ Native Applications vs Cross Platform

- ▷ Implementation-Specific Functionality
- ▷ Added Maintenance

■ Cloud Provider - AWS IoT

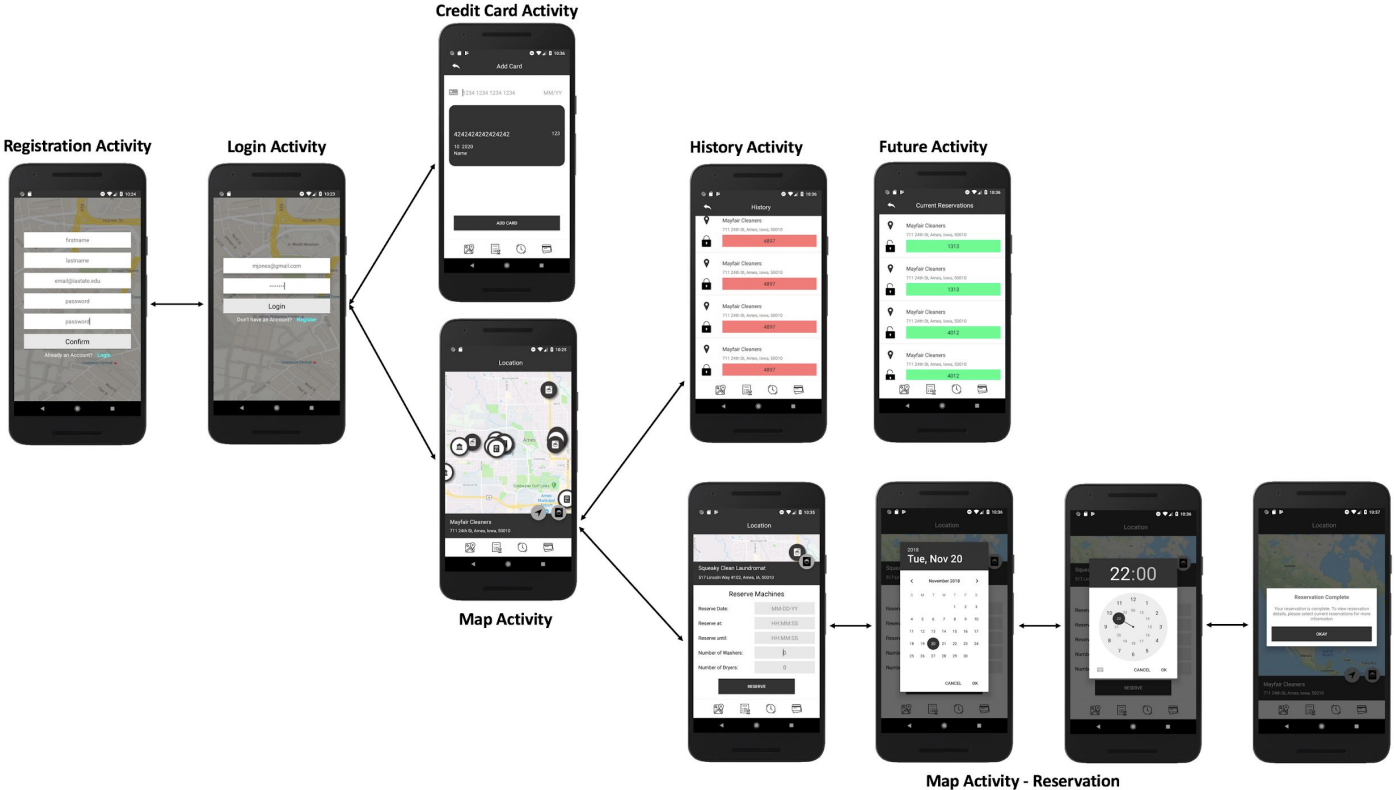
- ▷ Clients' Request
- ▷ Prior Experience

5

System Implementation

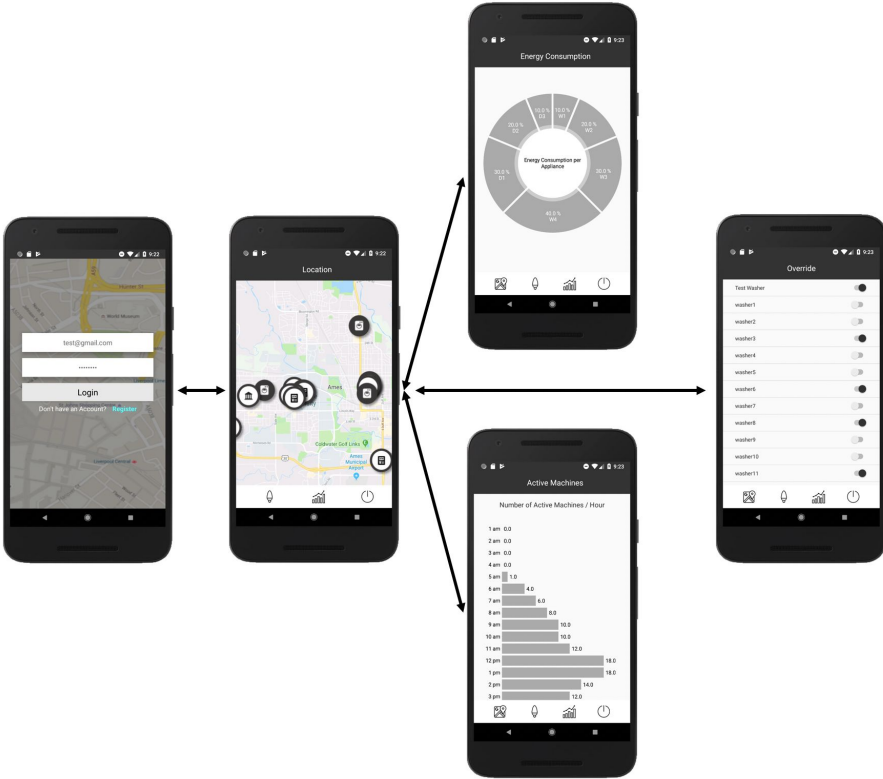


Customer Interface





Admin Interface





Mobile Implementation

SDKS

- Stripe SDK (Payment Transaction Service)
- Google Maps SDK

Libraries

- MPAndroidCharts
- Charts



Backend Design

Web Server

- Spring Boot Web Server (SBWS)

Cloud Service

- Cloud Hosting (AWS)

Database

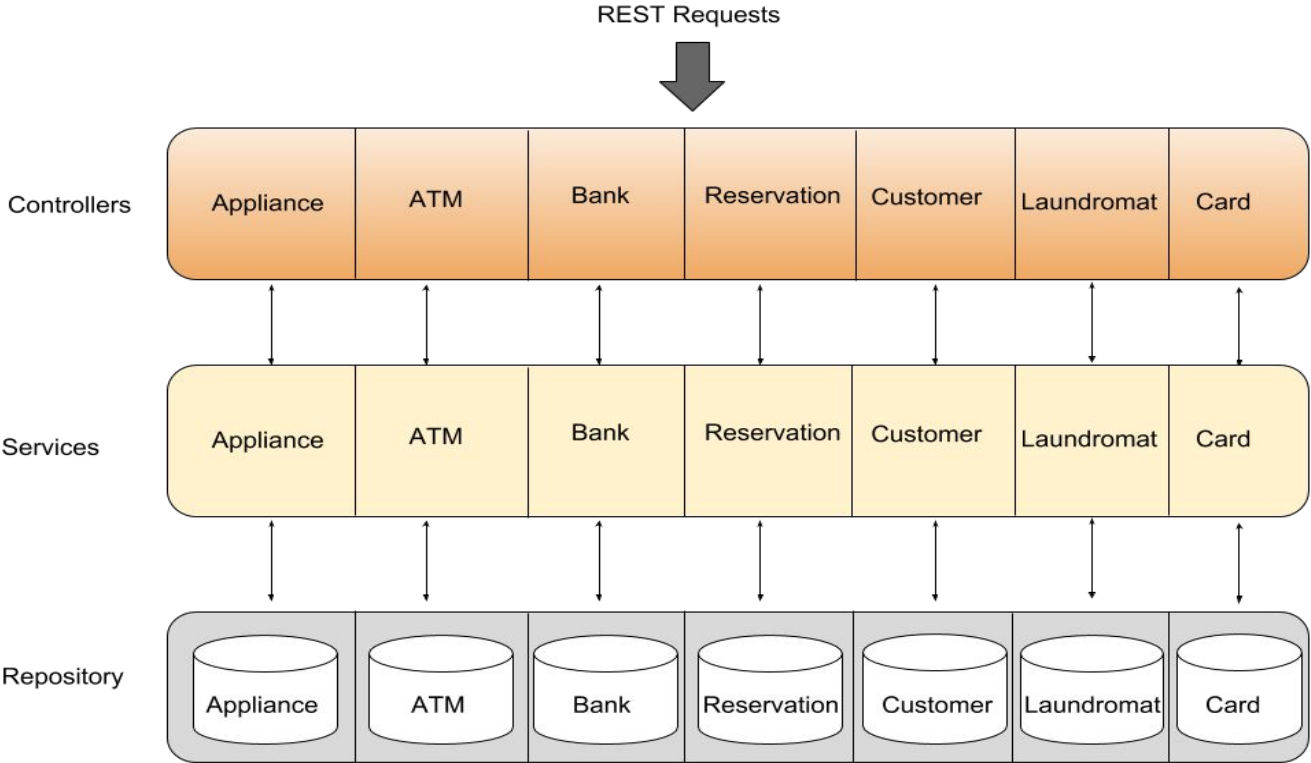
- MySQL External Database

Communication

- MQTT Integration



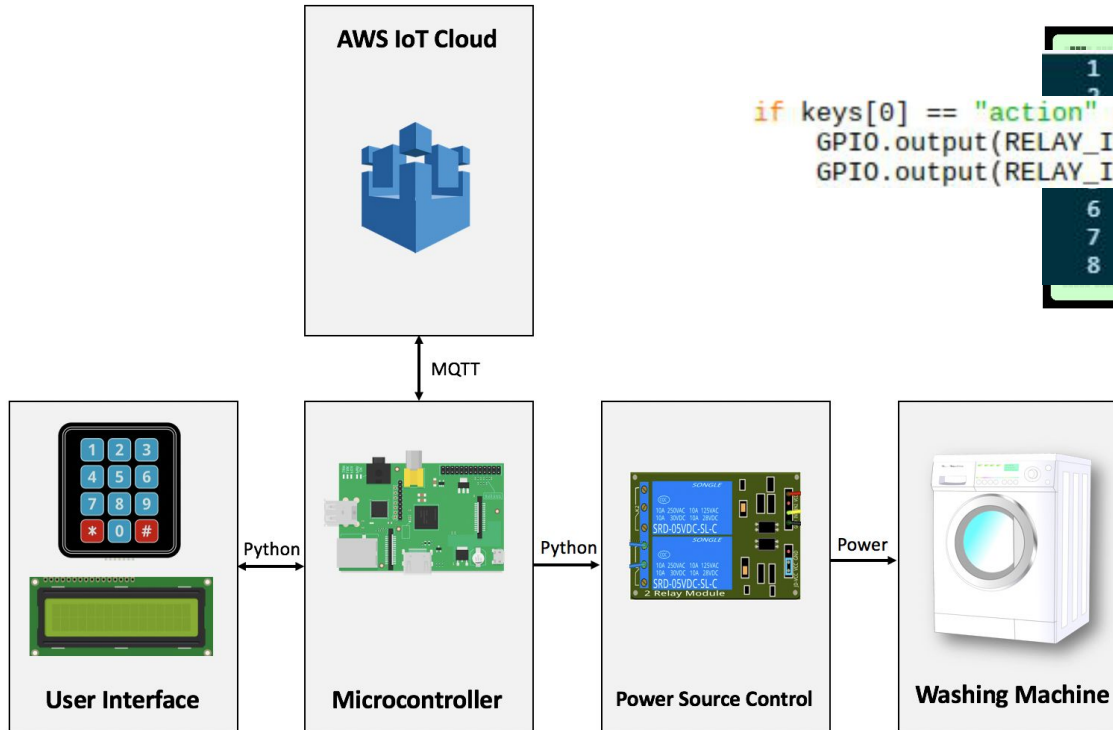
Backend Diagram



- SBWS
- AWS
- MySQL
- MQTT



Hardware Concept Diagram

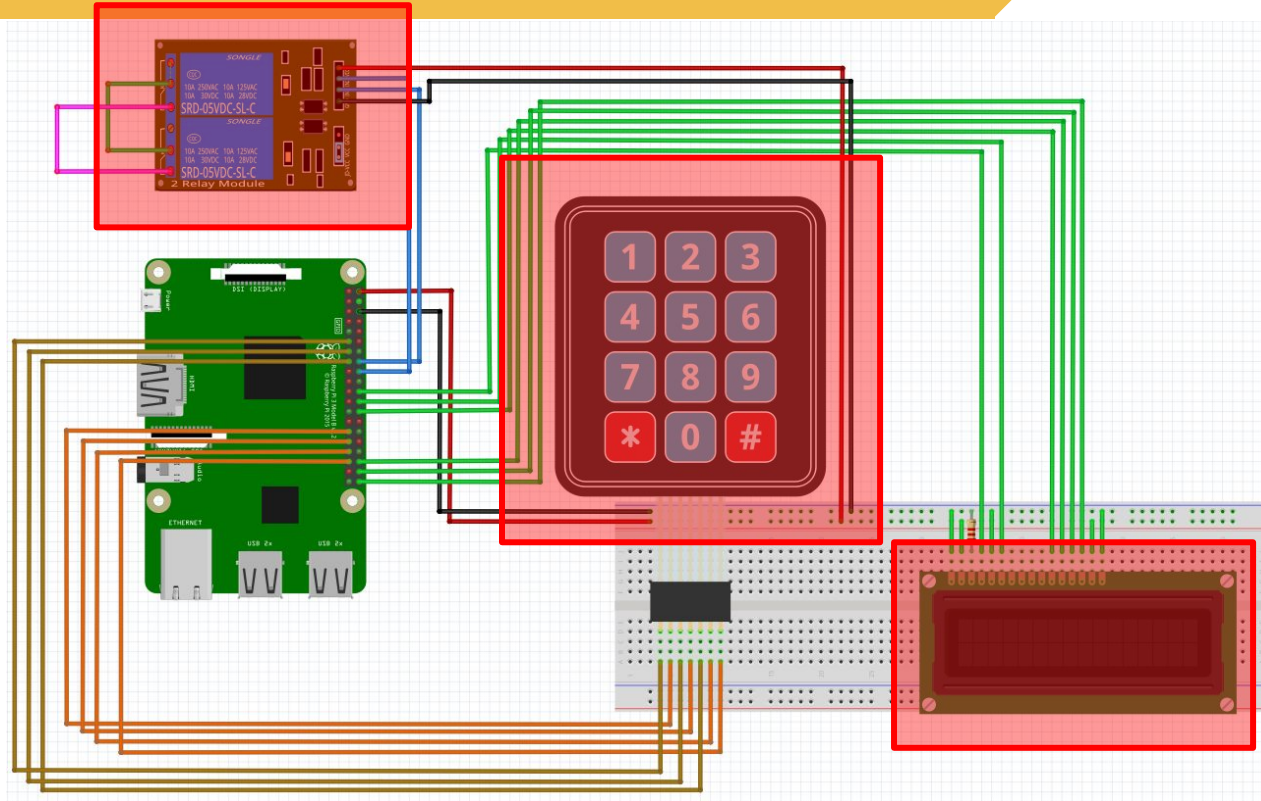


```
if keys[0] == "action" and data['action'] == "on": #Power on
    GPIO.output(RELAY_IN_1, GPIO.LOW)
    GPIO.output(RELAY_IN_2, GPIO.LOW)
```

```
1 {
2   "machine": "1", "code": "123456"
3 }
4
5
6   "action": "on", "time": "3600"
7 }
8
```



Electronic Diagram



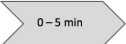
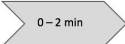
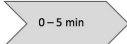
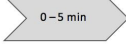
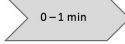
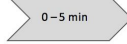
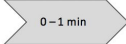

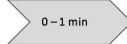
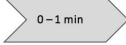
- RPi 17/40 pins
- 6 → LCD
 - 7 → Keypad
 - 2 → Relay
 - 2 → 5V Power

6

Testing



Mobile Usability Testing

Usability Test		
John Fleiner, Ben Young	jfleiner@iastate.edu benyoung@iastate.edu	Name Date
Product Under Test IoT Remote Monitoring Application for Commercial Appliances	Test Objective To verify and validate the user flow of the mobile application by making sure The steps to create and view a reservation are clear and concise	Business Case The test will address how likely a user is to complete a reservation with the mobile application
Procedure Step 1: Create an account  0 – 5 min	Step 2: Login  0 – 2 min	Step 3: Enter Credit Card info  0 – 5 min
Step 4: Select a Laundromat  0 – 5 min	Step 5: Open the reservation form  0 – 1 min	Step 6: Submit reservation  0 – 5 min
Step 7: Confirm Reservation  0 – 1 min	Step 8: View current reservation  1 click	Step 9: Enter reservation code  0 – 1 min
Step 10: Turn on appliance  0 – 1 min		

Summary

- On Average 7/9 Tests Originally Passed

Detailed Results

- Please refer to pages 64 - 67 for more detail



Mobile Unit Testing

is_reservation_num_appliances_valid()	Given a request for 1000 machines, verify that the application displays a warning and prevents a reservation from being created	False	False
is_reservation_valid_card()	Given a valid reservation, but an invalid credit card, verify that the application displays a warning and prevents a reservation from being created	False	False

Summary

- 21/28 Tests Originally Passed
- 75% Success Rate

Detailed Results

- Please refer to pages 67 - 71 for more detail



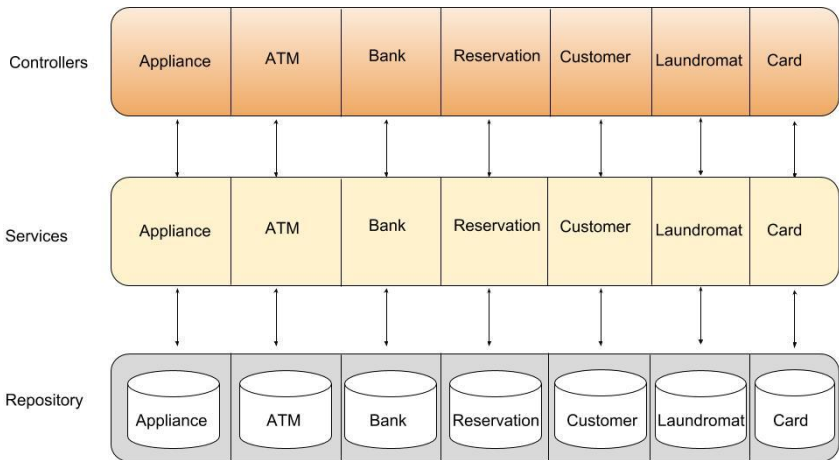
Backend Testing & Results

Completed JUnit Testing

- Service Layer
- Repository Layer

Runs: 5/5 Errors: 0 Failures: 0

```
com.greinerjennings.iamonitoring.service.ReservationServiceImplTest [Runner: JUnit 4] (0.011 s)
├─ testGetReservationHistory (0.011 s)
├─ testGetReservationsBetweenDates (0.000 s)
├─ testGetReservationsBetweenDatesAtLocation (0.000 s)
├─ testGetFutureReservations (0.000 s)
└─ testAddReservation (0.000 s)
```





Hardware Testing

Onboard Scripts Testing

- Python Scripts
- Shell Scripts

Circuit Functionality Testing

- Relay Module
- LCD Screen
- Keypad



Lessons Learned & Client Feedback

Lessons Learned

- Soldering: Delicate Work
- Many UI Iterations

Client Feedback

- Pleased with Prototype
- Un-concerned of Scalability

7

Prototype



Prototype Demo





Future Work - Commercialization

Commercialization

- Data Privacy Enhancements
- AWS Scalability

Future Work

- Surge Pricing
- Hardware Usage Analytics
- Multi-Location Analytics



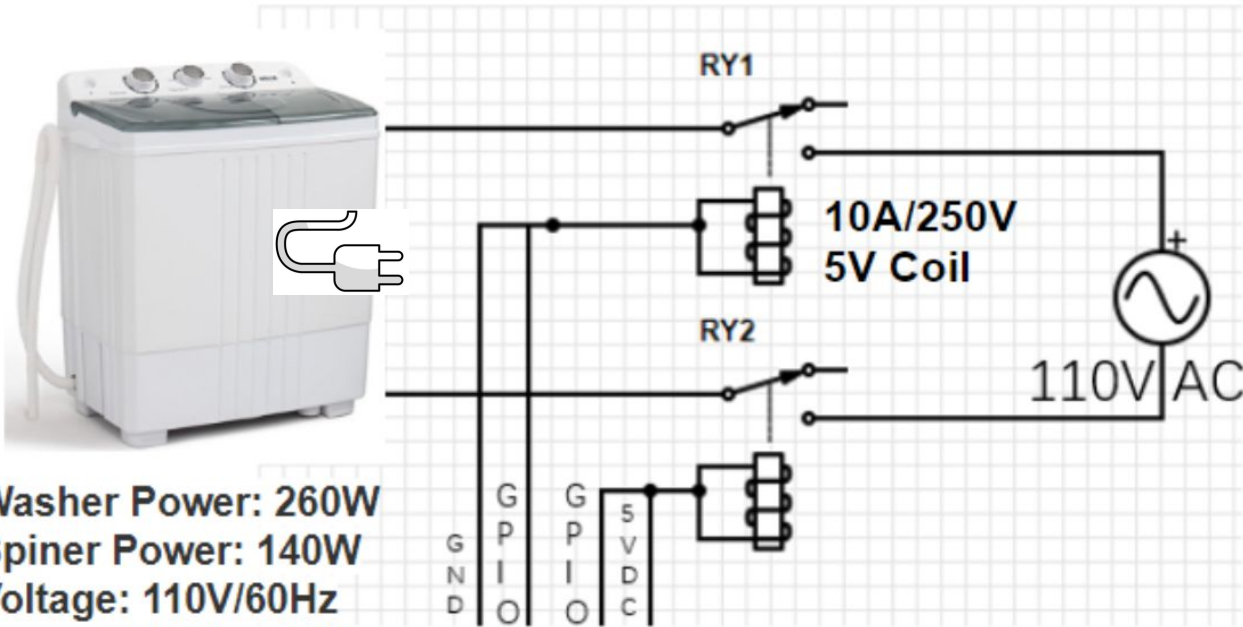
THANKS!

Any questions?

sddec18-17@iastate.edu



Circuit Diagram



Washer Power: 260W
Spinner Power: 140W
Voltage: 110V/60Hz
Current: 1.2A ~ 2.2A