IoT Information Display

Transcript

1. **Internet of Things (IoT) Information Display**
   1. Hello everyone!
      1. Thanks for being patient
      2. I’m Jacob, and I’m joined by \_(Ben)\_
      3. We’re here to show you the Internet of Things (IoT) Information Display
      4. I hope that you’ll find it as interesting and exciting as we do
2. **Contents**
   1. The contents are mostly divided into
      1. an introduction,
      2. a core,
      3. and some finishing thoughts
   2. First, let’s get to the actual introduction
3. **Introduction**
   1. Our project is a
      1. wall-mounted,
      2. touch-screen display,
      3. with internet connectivity
   2. This would ideally go outside a professor’s office
      1. This then could help with student-professor communication
   3. The device displays
      1. The weather
      2. A professor’s google calendar
      3. Announcements or memos posted through twitter
      4. And advertisements or just a slideshow of pictures
   4. But these are lot of terms to describe the device, so we’ll just call it the **IoT Information Display**
4. So for the **Concept,**
   1. The goals of the display include:
      1. Having an Attractive, user-friendly interface
      2. Pulling daily calendar data
      3. Pulling current and forecast weather data
      4. Showing announcements or short memos
      5. Having advertisements or a collection of pictures
      6. And an Availability ‘Status’
   2. \*point on screen to describe each aspect of the picture\* <quickly>
      1. Announcements shown through twitter
         1. ‘Status’ icon
      2. Weather data pulled from gov website
         1. Temperature
         2. Weather conditions
      3. Calendar from your google account
   3. We’re Also using the reliable technologies of:
      1. Python 3.6
      2. Openweathermap.org for Python
      3. Google Calendar API for python
      4. Twitter API for python
5. So with that introduction, let’s take a look at the previous project iteration, which Ben will describe to you now; **Previous Work (2016-2017)**

~~

~~

1. **Similar Projects**

~~

~~

1. **Research**

~~

~~

1. **Components - Hardware**

~~

~~

1. **System-Level Diagram**

~~

~~

1. **Mounting Diagram**

~~

~~

1. **Components – Software**

~~

~~

1. **Functionality - Modes of Operation**

~~

~~

1. **Functionality - Modes of Operation Diagram**

~~

~~

1. \*yup\* and so for the **Functionality** actual device:
   1. To the left you have your google calendar
      1. Synchronizes with your google account
   2. In the middle is the weather information
      1. The icon in the middle,
      2. The background,
      3. Along with the Temperature;
      4. All update with newly-pulled weather data
   3. And on the right is the twitter announcements
      1. With availability ‘status’ icon at the bottom, in the middle
   4. And again, all of these are updated
      1. Every 10 minutes
   5. When someone taps / touches one of these parts of the screen, it pulls up more information in a window for that application (seamless transition)
2. **Functionality – Google Calendar**
   1. Here we have the google calendar window when touched;
   2. Any events that you have created on your google account will show up here
   3. Displays 4 days of events
   4. And the Colors are based on the original creation
      1. So if you made the event and changed the color to ‘banana’ or ‘tomato’;
      2. those colors correspond here,
         1. in a slightly darker shade for increased visibility

~~

~~

* 1. \*yes and\* the Setup guide for the API is on our project website

1. **Functionality – Weather**

~~

~~

1. **Functionality – Twitter**
   1. Displays a longer feed of tweets from the user’s timeline
   2. Availability ‘status’ icon is updated in the twitter feed
      1. \*available
         1. Update to green circle
      2. \*away
         1. Update to yellow circle
      3. \*busy
         1. Update to red circle
      4. \*hide or \*offline
         1. Hides the availability ‘status’ icon
            1. For people who don’t want to use or show it
            2. Or When you’re at home and not at the office

~~

~~

* 1. \*yup and\* the Setup guide for Twitter API is on our project website

1. **Functionality – Advertisements / Pictures**
   1. This Cycles through a series of pictures
   2. Will automatically default to this version after a set period of time
   3. Can go back to home screen with simple touch on the screen
2. **GUI Process Diagram**

~~

~~

1. **GUI Process Diagram (1)**

~~

~~

1. **GUI Process Diagram (2)**

~~

~~

1. **Parts List**

~~

~~

1. **Schedule (Fall 2017)**

~~

~~

1. **Schedule (Spring 2018)**

~~

~~

1. **Division of Labor**

~~

~~

1. **Future Direction**
   1. Preface:
      1. Project goal was to create a great baseline for future development
      2. Groundwork for more app implementation
      3. Starting point for improved graphical and more eye-appealing design
      4. Something a hobbyist can be inspired by
   2. Geofencing via cellular data
      1. This was in our original design
         1. But we decided to allocate our time to the core features
      2. I deemed it as somewhat controversial
         1. Being tracked
         2. Having to carry phone around
         3. Having to download a specific appjust to be tracked
      3. Minor locational issues
         1. Could display available but in wrong location
   3. Automatic offline or hide status
      1. Thought Brought up by IAB presentation
      2. Availability status provides unique problems
         1. If used, have to actively update
      3. Frequent time updates would need a dedicated call timer set on a shorter frequency
   4. Voice Capabilities
      1. Something Dr. Ahn asked about in the previous presentation
      2. Talked about implementing something like amazon alexa
      3. With being a touch screen:
         1. Dirt, fingerprints
         2. Germs D:
   5. Creating new tweets or calendar events straight on the device
      1. Straightforward
      2. Generically you want more customization when doing this and it’s hard to validate the reason to spend so much time implementing it when it’s already optimized on the official website or app
   6. Show pictures via twitter
      1. Something the url version of a simple developer twitter url could handle much easier
      2. Much more difficult for python and appJar
         1. Especially with drawing everything with appJar
   7. Sensors
      1. Door
         1. Sense whether professor is in the office
      2. Camera
         1. Capture students or other users’ who come looking for help
      3. Somewhat controversial as well
         1. Being tracked
   8. Online Expansion
      1. Remote use and view
      2. Get notified on certain updates in an easy-to-read format
      3. Somewhat pointless when you can just straight view google calendar and twitter straight from home
   9. Email and or text instructor from display
      1. Problem with spamming
   10. Appointment scheduler
       1. Whole separate interface
       2. Great addition
          1. But takes time and implementation of 3rd party resources potentially
             1. Youcanbookme api
          2. Helps with the core reason why students go to professors’ offices in the first place
   11. Even more potential
       1. Hopefully, as time goes on, and with a new set of eyes and new inspirations, this project is a great foundation to expand upon and build some marvelous creations
2. **References**
   1. A lot of the api references and guides can be seen in these links
   2. Also, at the bottom is our project website
      1. You can go there for the two google and twitter api implementation guides
         1. If you’re looking to do something like this yourself 😊

FIN