Chandra’s Example Project

- Online data visualization for IoT sensor data

UC Santa Barbara

Edge Cloud

= Virtual Server
(Store/Forward Data to cloud)

IoT data

DockerHub

Virtual Server @ UCSB
+ Docker, Postgres,
(web frontend + backend)

Public/Private Cloud

IoT data

User login

User auth

Dynamic data vis

Firebase
Chandra’s Example Project

- Online data visualization for IoT sensor data

- Technology Investigations:
  - Virtual Linux server (Edge and cloud) + Docker/Dockerhub
    - Setup and configuration
  - User login support: front end and backend
    - Web front end (html, css, javascript, 3rd party/oauth?)
      - Bootstrap, material design for React w/ bootstrap
    - Backend (firebase)
      - When you should and shouldn’t use firebase
      - Link bootstrap to firebase
  - Time series graphs from local data: Chart.js
  - Dynamic time series graphs
    - GraphQL + Postgresql + Chart.js
  - Connect logged in user to authenticate data access
  - Ingress data to edge cloud server
  - Forward edge data to cloud postgresql DB
Chandra’s Example Project

• Online data visualization for IoT sensor data
• Sprint 1 plan: *(Sprint 1 is 9 days total)*
  – AWS EC2 virtual server tutorial with persistent volume storage
    • **test**: start/stop server, (un-)mount volume, ssh, config Docker/ web backend (below)
  – Docker tutorial: configure/deploy server image: **test**=all team members laptops and EC2 virtual server
  – user login support: front end and backend
    • Bootstrap/React tutorial: **test**=deploy simple website, be able to change it
      – Bootstrap, [material design for React w/ bootstrap](https://material.io)
    • [firebase](https://firebase.google.com) tutorial, link to bootstrap: **test**=simple login page
    • Investigate how to share info on logged in user to remote service
  – Web page static graph: [Chart.js](https://chartjs.org) tutorial: **test**=website with graph w/ fake data (hand coded in javascript)
  – Dynamic time series graph
    • Tutorials + [Integration tutorial: GraphQL Engine + Postgresql+ Chart.js](https://www.graphql.org)
    • **Test**: website that is dynamically updated when new data is added to db
  – PRD work (pbm, innovation, detailed core advance, background, sys arch picture, technologies list), **test** = review by team mate
Chandra’s Example Project

- Online data visualization for IoT sensor data
- Sprint 1 plan: (Sprint 1 = 9 days – breakdown is below in red)
  - AWS EC2 virtual server tutorial with persistent volume storage
    - test: start/stop server, (un-)mount volume, ssh, config Docker/ web backend (below)
  - Docker tutorial: configure/deploy server image: test=all team members laptops and EC2 virtual server
  - user login support: front end and backend
    - Bootstrap/React tutorial: test=deploy simple website, be able to change it
      - Bootstrap, material design for React w/ bootstrap
    - firebase tutorial, link to bootstrap: test=simple login page
    - Investigate how to share info on logged in user to remote service
  - Web page static graph: Chart.js tutorial: test=website with graph w/ fake data (hand coded in javascript)
  - Dynamic time series graph
    - Tutorials + Integration tutorial: GraphQL Engine + Postgresql+ Chart.js
    - Test: website that is dynamically updated when new data is added to db
  - PRD work (pbm, innovation, detailed core advance, background, sys arch picture, technologies list), test = review
    by team mate
  - Trello setup for Sprint 1
Chandra’s Example Project

- Online data visualization for IoT sensor data
- **Sprint 1 backlog (will move to sprint 2)**
  - Connect logged in user to authenticate data access
  - Setup/configure edge server (using docker container for portability)
  - Ingress data to edge cloud server
  - Forward edge data to cloud postgresql DB
  - Features as determined by requirements analysis: user stories and use cases (more on this next Monday)
- Add these to new Sprint 2 board (Backlog)

- [Create burndown](#)