Project Guidelines

Chenyang Lu
Steps

1. Come up with your favorite topic
2. Form a team
3. Proposal: propose a design and plan
4. Analyze and implement your solution
5. Evaluate your solution
6. Demo 1, 2 and Final Demo
7. Write a technical report
Topics

- Three students per team
  - Build a system
  - Write a paper
  - Demo to the class

- Option A: Build a real-time embedded system
  - Explore how to achieve and optimize real-time performance

- Option B: Build IoT systems based on cloud and edge
  - Front end: Raspberry Pi…
  - Cloud and edge: storage, analytics, Alexa, notification

- Address latency
  - Measure latency (mean, variance, tail)
  - Evaluate the impacts of different configurations (e.g., local vs. edge vs. cloud) on latency.
Start Early and Work Often!

- Choose topic
- Put together a team
- Meet every week to coordinate
- A lot of work (and fun) throughout the semester!
Teaming

- Everyone should be in a **three**-member team
  - Need special approval from TA for a different size

- Use **Piazza** to “Search for Teammates”

- Email Eric your team members by **1/27**
  - One email per team

- We will help make sure everyone has a team.
Proposal Presentation

- In class on 2/3

- **5 min** per group
  - 4-min talk + 1-min Q&A (hard deadline)
  - 2-3 slides
  - Rehearse over Zoom
  - **Turn on your video** during your presentation

- Your **elevator pitch!**
Written Proposal

- One proposal/team, one page
  - Team members
  - Overview of project
  - Responsibilities of each member
  - Equipment

- Submit on Canvas by 2/3 (11:59pm)
  - Written proposal
  - Presentation slides
Demo I

- **In person** on 3/1 and 3/3
- **10 min** per team
  - 9-min demo + 1-min Q&A
- **Must show something** **real**
- **Submit slides before the class of your demo**
Demo II

- **In person** on 3/29 and 3/31

- **10 min** per team
  - 9-min demo + 1-min Q&A

- Substantial progress towards final demo

- Submit slides before the class of your demo
Final Demo

- Final demo (in person)
  - 4/28, 1pm-3:15pm
  - Busch Hall 100

- 10 min per team
  - 9-min demo + 1-min Q&A
  - All should attend the entire session. It’ll be fun!

- Check out room and **test** your demo
  - 4/26, 1pm-2:15pm
  - Busch Hall 100

- Submit before class: slides, backup video if needed
Final Report

- Submit on Canvas by **5/5, 11:59pm**.

- **Report**
  - Format and style: follow conference papers in the reading list
  - 6 pages, double column, 10 pts font
  - Use templates on the class web page

- **Materials**
  - Web page
  - Slides of your final presentation
  - Source code
  - Documents: README, INSTALL, HOW-TO-RUN
  - Video
Abstract

1. Introduction
2. Goals and Requirements
3. Design
4. Implementation
5. Experiments
6. Related Works
7. Lessons Learned
8. Conclusion and Future Work
Peer Review

- For fairness in group projects.

- Email me on 5/5
  - Percentage of contributions of each team member.
  - Brief justification.
Car Informatics in the Cloud

- Pull real-time OBD data from a car
- Upload to the Cloud and display stats at real-time

BY Ethan Vaughan, Frank Sun, and Adith J. Boloor
Follow-Me Music
Spice Bot: Spice-Blend Automation

- 3D-Printed Prototype
- Voice-Control-Interface
  - Amazon Echo
- Actuator Control
  - Raspberry Pi
- Control Command Interpretation
  - AWS IoT

BY ALEX HERRIOTT, QUOC NGUYEN, RAYMOND JONES
Smart Lock

- Remote doorway system
  - Live video
  - Arrival (motion) detection
- Web application
  - Node.js server on an EC2 instance
  - Live video via ssh tunnel
  - Engage/disengage lock

BY Charles Ahrens Feldman, David Ayeke, and Steven Bosch
Explore Edge Computing

벽크마크 및 최적화 latency with edge services

AWS IoT Greengrass
https://aws.amazon.com/greengrass/
Real-Time Edge Cloud

- Advanced control and analytics → **edge cloud** shared by applications
- Interacting with physical environment → **real-time** performance
- Example: automotive systems
  - Shared platforms: ~100 ECUs → ~10 multicore processors (**edge cloud**)  
  - Multiple virtualized systems

**Real-time performance guarantees in edge clouds**

Xen

- **Xen: type-I, baremetal hypervisor**
  - Domain-0: drivers, tool stack to control VMs
  - Guest Domain: para-virtualized or fully virtualized OS

- **Scheduling hierarchy**
  - Xen schedules VCPUs on PCPUs.
  - Guest OS schedules threads on VCPUs.
  - Credit scheduler: round-robin with proportional share.
RT-Xen

- Real-time schedulers in the Xen hypervisor.
- Provide real-time guarantees to tasks in VMs.
- Started as a course project and grew into a major research effort.

Impacts

- Transformed compositional scheduling from theory to virtualized platforms.
- Produced the real-time deferrable server (rtds) scheduler in the Xen hypervisor.
Logistics

- TA for projects: Hanyang (Eric) Liu
- Email TA or me for appointments to discuss ideas
- All work will be submitted on Canvas