CS Capstone Design

Demo Flight Plan

TEAM: HelloWorldByMe

Overview: The main purpose of the "Technical Demos" is to very clearly communicate the extent to which the team has identified key challenges in the project, and has proven solutions to those challenges. Grading is based on how complete/accurate the list of challenges is, and how convincingly and completely the given demos cover the given challenges.

This template is fleshed out by the team, approved by CS mentor, and brought to demo as a grading sheet.

Risky technical challenges

Based on our requirements acquisition work and current understanding of the problem and envisioned solution, the following are the key technical challenges that we will need to overcome in implementing our solution:

C1: Cross-communication between databases.

Challenge: Establishing seamless and reliable connections between different databases is important to ensure client data and other relevant information is safely managed.

Proof: We will demonstrate database cross-communication by implementing a Python-based solution where a "master" database will be set up that will hold references to other databases. From there, the switching between databases will be done with a Python cursor object where they will be executing a query to retrieve data.

C2: Real-time messaging between two entities.

Challenge: Real-time messaging is crucial when allowing Navigators and service providers to coordinate efficiently without delay. This requires a solution that will be able to handle concurrency and data consistency.

Proof: A live demonstration of two clients sending and receiving messages from each other using a web socket will be shown.

C3: Pinpointing the location using manual input.

Challenge: Accurately mapping a manually added location is essential for fulfilling one of the requirements, which is to be able to allow current coordinates to be stored/recorded.

Proof: To fulfill these requirements, a demonstration of manual input of coordinates and converting it to a location using a geospatial API will be done.

Challenges covered by demos:

In this section, we outline the demonstrations we have prepared, and exactly which of the challenge(s) each one of them proves a solution to.

Demonstration 1: Database Cross-Communication

Challenges addressed:

C1: Cross-communication between databases.

Flight Plan: Step-by-step overview of the demo

1. Initialize master database.
2. Establish a connection between 2 instances.
3. Use the Python cursor object to switch between the databases.
4. Execute queries on the target database.
5. Return results to the master from the target database.

Demonstration 2: Real-Time Messaging Interface

Challenges addressed:

C2: Real-time messaging between two entities.

Flight Plan: Step-by-step overview of the demo

1. Initialize two client applications by connection via WebSocket.
2. Send a message from one client to another.
3. Display the second client receiving the message.
4. If sudden disconnection happens, simulate a reconnection.

Demonstration 3: Manual Location Input

Challenges addressed:

C3: Pinpointing the location using manual input.

Flight Plan: Step-by-step overview of the demo

5. Present a form for manual input of location.
6. Use a geocoding API to demonstrate the pinpointing of the location.
7. Display mapped location.

Other challenges recognized by not addressed by demo:

If there were challenges you listed earlier that were *not* covered by a demo, list here. This will hopefully be a short list…but better to be clear about where you are. If you have items here, you could list (if applicable) any pending plans to reduce these risks.

- role based access
    - forms for roles (data collection)
- user authentication??
-