Submitted by Akshay Joshi & Smita Bhattacharya

Architectural Thinking Assignment 4

1. **Formulate 3 high-level user stories including an acceptance test**
   i) **User Story 1:** As an End User, I want to query for next day's weather forecast so that I am aware of the changes in weather for my current location
      - **Acceptance Test:** If an active internet connection & GPS location of the user is available and the user has already logged-in into the system, whenever the user makes a weather forecast query, generate weather forecast results for his current location and deliver it either by voice or text

   ii) **User Story 2:** As a Manager in an organization, I want to book a specific meeting room so that I can schedule a meeting with my team
      - **Acceptance Test:** If an active internet connection is available and the user has already logged-in into the system, whenever the Program Manager gives a command to the system to book a vacant meeting room, the system should check if the requested room is vacant or not at that particular time and schedule the booking accordingly and also send the response back to the user via a text/voice alert

   iii) User Story 3: As an Infrastructure Administrator, I want to query if the Fire Extinguisher system inside the building is online so that if in case of a fire emergency it is handled correctly
      - **Acceptance Test:** If an active Internet/LAN connection is available and the user has already logged-in into the system, whenever the Infra Admin queries if the Fire Ext is online, the system should check if the fire extinguisher tanks are full or not and also the smoke detection sensor is calibrated correctly. If yes, send the response back to the admin via a text/voice alert

2. **Check each story based on the INVEST acronym and summarize your considerations**
   i. User Story 1: The story is independent & negotiable. But, not estimable. Because, we are not aware about other features of weather forecast.
   ii. User Story 2: This story is compliant to all the properties. Because, we can implement this story in any sequence and is also negotiable and estimable since there is no specificities exactly mentioned for implementation and we can estimate the time and effort needed to implement.
   iii. **User Story 3:** This story is compliant to all the properties. Because, we can implement this story in any sequence and is also valuable, estimable & small, since this is a crucial requirement for Infrastructure management and we can estimate the time and effort needed for implementation.

3. **How do you ensure that the acceptance test is measurable?**
   We can ensure the measurability of acceptance tests by following these steps:
   - Writing Acceptance tests which demonstrate a single goal
   - Writing Acceptance tests which does not depend on the results of other tests
   - Broad acceptance test/criteria make a user story vague. Hence, we can write effective acceptance tests which outline the scope of work so that the developers can plan and estimate their effort properly
   - Ensuring the acceptance tests are not cluttered with implementation details (keeping it abstract i.e., showing what happens & not how). Else these acceptance tests might become brittle, inflexible and not easily understandable
   - Create a set of acceptance tests for each user story using positive/negative scenarios and also consider the range of input/output bounds
   - We can include the 'rule' and a set of related acceptance tests for additional clarity (This shows the acceptance tests which illustrates that particular rule)

4. **Think about how you will implement the requirement described by the user story. Write a tentative version of the corresponding use case**
   Firstly, Use cases help explain how the system should behave and in the process, they also help brainstorm what could go wrong. They provide a list of goals and this list can be used to establish the cost and complexity of the system. Teams can then negotiate which requirements can be considered and built.
   So, once we have negotiated about the requirements from the use cases, we can pick one and start implementing it by following these steps:
   i. Iteration 1: Identify Initial requirement technology stack
   ii. Identify an Architectural overview
   iii. Model enough to get good estimates and also plan the work for the current iteration
   iv. Involve the stakeholders and discuss specific implementation issues
   v. Collect evolved requirements, if any
   vi. Implement the requirement using a test-first approach
   vii. Repeat the same steps for n Iterations

[Please turn the page over!]

Tentative version of a use case for User Story 1:

Attribute	Description
Goal of the Use Case	Generate weather forecast report for a specific location
Actor(s)	End Users/Public
Pre-conditions	{Active Internet Connection} {Current GPS location of the user} {User - authorized to log-in} {Sufficient storage space in the handset}
Basic Flow	1. User utters the wake word of the system (ex: Ok Google) 2. If there is an GPS & Internet connection available, the system is triggered on hearing the wake word/explicit application launch 3. System displays the list of available functionalities 4. User send a weather forecast query through voice or by explicitly typing "How is the weather" 5. Once the query is sent, system will triage the user's current location (latitude & longitude) 6. Once the location is determined, system will contact the weather service provider (Ex: Accuweather) and pass the location parameters 7. The weather service agent retrieves the weather data for the specified location 8. After the agent has successfully retrieved the data, the same will be sent to our intelligent system (perhaps as a XML, JSON file) 9. Now, the system would parse the and prep the results as per user preferences 10. The query is stored in Context Parser for future references 11. The NLP component inside the system receives these results and coverts them into speech fragments and also display the same results on the handset's screen 12. Users acknowledges these results 13. Based on the satisfaction of results, user would provide feedback to the system
Alternative Flow 1	1. User utters the wake word of the system (ex: Ok Google) 2. If there is no Internet connection available 3. The system triggers an error: "Please connect to the Internet"
Alternative Flow 2	1. User utters the wake word of the system (ex: Ok Google) 2. If there is Internet connection available. But, GPS is not turned 'ON' 3. The system triggers an pop-up: "Please turn ON location services"
Alternative Flow 3	1. User utters the wake word of the system (ex: Ok Google) 2. If the user is not authorized/not logged-in 3. System throws an error: "Please Register/Login to continue"
Post conditions	Once the above use case has been executed successfully, system checks the Context library and recommends any related tasks the user might be interested in

Use case specific non-functional requirements	1. The response of the system should be quick (less than 5 ms) 2. The forecast results generated should match for the current location accurately 3. The wake word/query detection of the system should be accurate
Special Considerations	System should consider User Preferences (like: Celsius or Fahrenheit) while generating the results
Creation Date	24.11.2019
Modification	Version 2 24.11.2019

5. **Disassemble each story as described on slide 24 and create one goal hierarchy showing potential dependencies between and inside the use cases**
    i. **User Story 1:** I want weather forecast report, to do this system sends the weather forecast info, to do this system generates weather forecast report, to do this system retrieves the weather forecast data, to do this the system check if the location given is correct or not, to do this I have to send my location query, to do this I have to send my location details, to do that I have to login to my account, to do this I have to register/authorize my account, to do this I have to enable Internet Connection

    ii. **User Story 2:** I want to book a meeting room, to do this system displays vacant room number, to do this system checks if room is vacant or not, to do this I have to send the room number, to do this I have to login to my account, to do this I have to register/authorize my account, to do this I have to enable Internet Connection

    iii. **User Story 3:** I want to check if Fire Extinguisher is online, to do this system shows if the fire ext. is online or not, to do this system checks if extinguisher tanks are full, to do this system checks if sensor is calibrated correctly, to do this I have to login to my account, to do this I have to register/authorize my account, to do this I have to enable LAN/Internet Connection

6. **Assign a color level to each goal as proposed by Cockburn. Make sure you use all color levels. If you find that that this is not possible, revise your user story and use case and refine your goal hierarchy from white down to indigo or even black**
    i. User Story 1:
        - **White (Cloud, Kite):** Display weather forecast, generate forecast report, retrieve weather data, check location correctness
        - **Blue (Wave):** Send query**,** send location data
        - **Indigo (Fish):** Login to the system, register account
        - **Black (Clam):** Setup network (IPv4/IPv6)

ii. **User Story 2:**

- **White (Cloud, Kite):** Book a meeting room, display if room is vacant or not
- **Blue (Wave):** Send query, send room number
- **Indigo (Fish):** Login to the system, register account
- **Black (Clam):** Setup network (IPv4/IPv6)

iii. **User Story 3:**

- **White (Cloud, Kite):** Display if Fire Ext is online or not, check if tanks are full, check if sensor is calibrated
- **Blue (Wave):** Send Fire Ext Online? query
- **Indigo (Fish):** Login to the system, register account
- **Black (Clam):** Setup network (LAN/WAN)

x-x-x