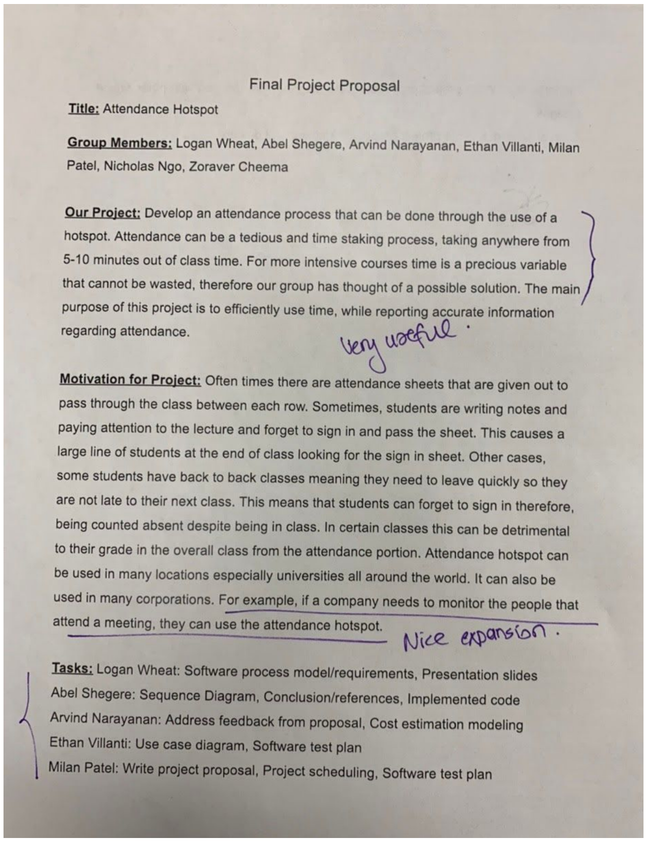
CS 3354 Software Engineering Final Project Deliverable 1

Attendance Hotspot

Arvind Narayanan, Zoraver Cheema, Logan Wheat, Abel Shegere, Ethan Villanti,
Nicholas Ngo, Milan Patel

1. Final Project draft description:



Final Project Proposal

Title: Attendance Hotspot

Group Members: Logan Wheat, Abel Shegere, Arvind Narayanan, Ethan Villanti, Milan Patel, Nicholas Ngo, Zoraver Cheema

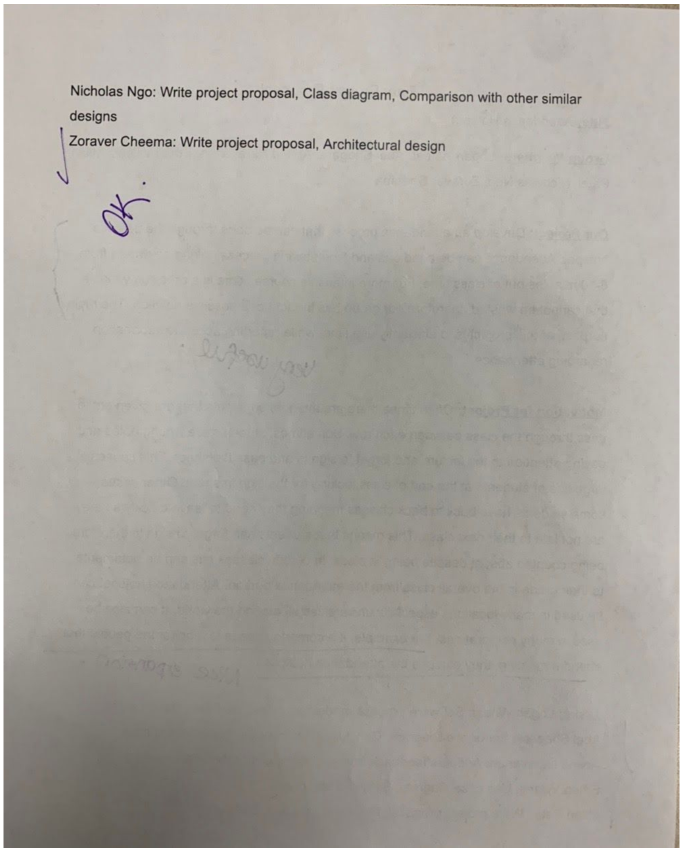
Our Project: Develop an attendance process that can be done through the use of a hotspot. Attendance can be a tedious and time staking process, taking anywhere from 5-10 minutes out of class time. For more intensive courses time is a precious variable that cannot be wasted, therefore our group has thought of a possible solution. The main purpose of this project is to efficiently use time, while reporting accurate information regarding attendance.

very useful.

Motivation for Project: Often times there are attendance sheets that are given out to pass through the class between each row. Sometimes, students are writing notes and paying attention to the lecture and forget to sign in and pass the sheet. This causes a large line of students at the end of class looking for the sign in sheet. Other cases, some students have back to back classes meaning they need to leave quickly so they are not late to their next class. This means that students can forget to sign in therefore, being counted absent despite being in class. In certain classes this can be detrimental to their grade in the overall class from the attendance portion. Attendance hotspot can be used in many locations especially universities all around the world. It can also be used in many corporations. For example, if a company needs to monitor the people that attend a meeting, they can use the attendance hotspot.

Nice expansion.

Tasks: Logan Wheat: Software process model/requirements, Presentation slides
Abel Shegere: Sequence Diagram, Conclusion/references, Implemented code
Arvind Narayanan: Address feedback from proposal, Cost estimation modeling
Ethan Villanti: Use case diagram, Software test plan
Milan Patel: Write project proposal, Project scheduling, Software test plan



Nicholas Ngo: Write project proposal, Class diagram, Comparison with other similar designs
Zoraver Cheema: Write project proposal, Architectural design

ok.

In our Final Project Draft description, we adequately described the objective for this project our group is working on. We did not receive any feedback from the professor regarding any issues with our draft, so we are assuming that our draft is sufficient enough to proceed with the project.

2. Delegation of Tasks

Logan Wheat: Software process model/requirements, Presentation slides
Abel Shegere: Sequence Diagram, Conclusion/references, Implemented code
Arvind Narayanan: Address feedback from proposal, Cost estimation modeling
Ethan Villanti: Use case diagram, Software test plan
Milan Patel: Write project proposal, Project scheduling, Software test plan
Nicholas Ngo: Write project proposal, Class diagram, Comparison with other similar designs
Zoraver Cheema: Write project proposal, Architectural design

3. Software Process Model Used

For this project our group will follow the waterfall model. This is because the project idea is mainly focused on one primary task, which is to take the attendance of the students that are in class, or employees at a certain event. Since the task is only to take attendance, the project most likely will not need to be flexible. Therefore, the use of the agile methodology or any repetition is unnecessary and can be seen as redundant. By using the waterfall method, the specifications and requirements would diligently gathered. This would be followed by a clear design to fit the specifications that are created. Once a design is created, it can then be implemented and tested. Once the testing of the finished product is completed, it can be deployed for use by customers. The team will maintain the released product and ensure no problems arise within the product.

4. A.) **Functional Requirements**

  - Log in with proper university or employee credentials. (If invalid credentials are entered give user error message and allow for retry)
  - Give options for available hotspots to connect to. (If none available then display proper message)
  - Once successful connection is established, allow for sign up to be done (Display success or error message).
  - Display upcoming classes or meetings for the day. (Possible sign up times)
  - Application will not allow sign in for classes or meetings that have not started yet.
  - Error message and sign out will occur if user disconnects from hotspot during class or meeting session

  B.) **Non-Functional Requirements**

  - Product requirements:
    - Usability requirements: The attendance hotspot should be available to any teacher or leader that needs it. This should be done through either an application utilizing on board network adapters, or even through the use of an external device.
    - Efficiency requirements:
      - Performance requirements: The application/external hotspot should quickly (relative) and consistently maintain connection and be able to connect up to 255 devices at the same time. More connectability should be available if offered in budget for network adapter.
      - Space requirements: The application/external hotspot needs to properly store the attendance information for each student correlating to the class, time, and professor. 1 Gigabyte should be more than enough space to store the application

        code, root user authentication, and attendance information for thousands of students across multiple different class times.

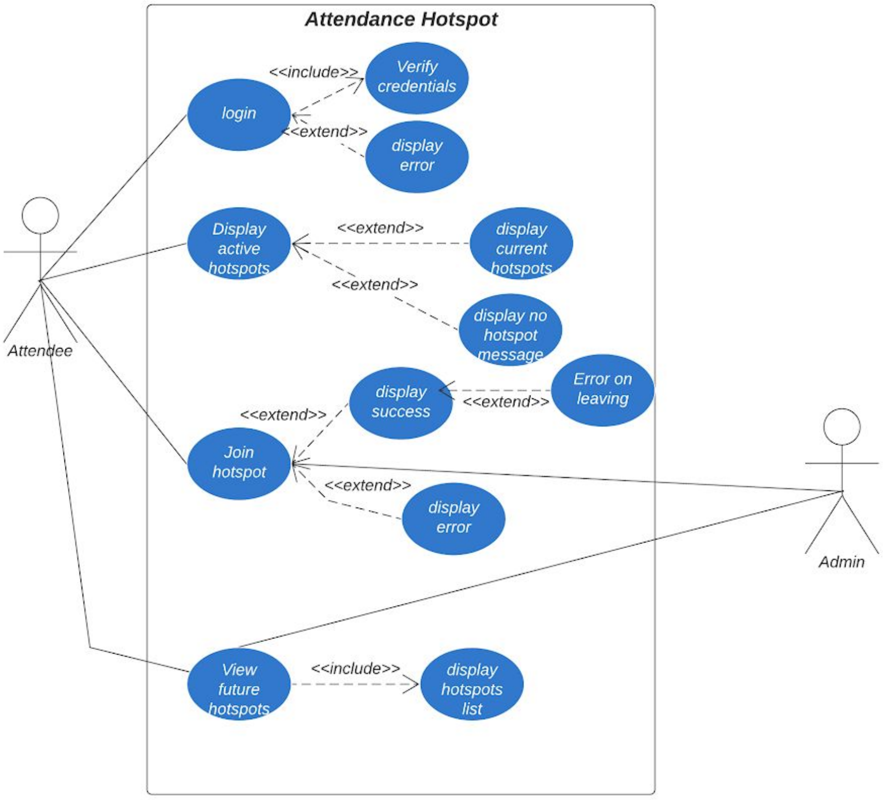
    - Dependability requirements: Downtime should not exceed more than 5 minutes during use hours (let's say, monday-sunday 8am - 10pm). If updates are necessary, they can roll through outside of use hours. Otherwise, the application/external hotspot should be dependable during the use hours.
    - Security requirements: Security should be a primary concern. Both the network data and onboard data with student information should be encrypted. The user should also have the option to use third party security measures, such ad Duo security used by the University of Texas at Dallas. The attendees information will also be protected.
- Organizational requirements:
    - Environmental requirements: Application/external hotspot should have access to internet in order to properly authorize credentials.
    - Operational requirements: Authorization should be made using credentials from university or employer, meaning that the application should also have access to those databases.
    - Development requirements: There will be different privileges depending on if you are primary user or attendee. This will allow for more organization within the database or on board information.
- External requirements:
    - Regulatory requirements: Each time the application/device is used, the primary user must be validated using proper credentials.
    - Ethical requirements: The application/device is intended to be used only for creating and maintaining data on attendance. The process will not breach IEEE ethical code of conduct.

- Legislative requirements:
    - Accounting requirements: The applications intent is to be used for attendance policies, therefore, the developers will not be held directly responsible if used other than intended. Security updated will take place in this event.
    - safety/security requirements: As stated before, security should be a primary concern. Both the network data and onboard data with student information should be encrypted. The user should also have the option to use third party security measures, such ad Duo security used by the University of Texas at Dallas. The attendees information will also be protected.

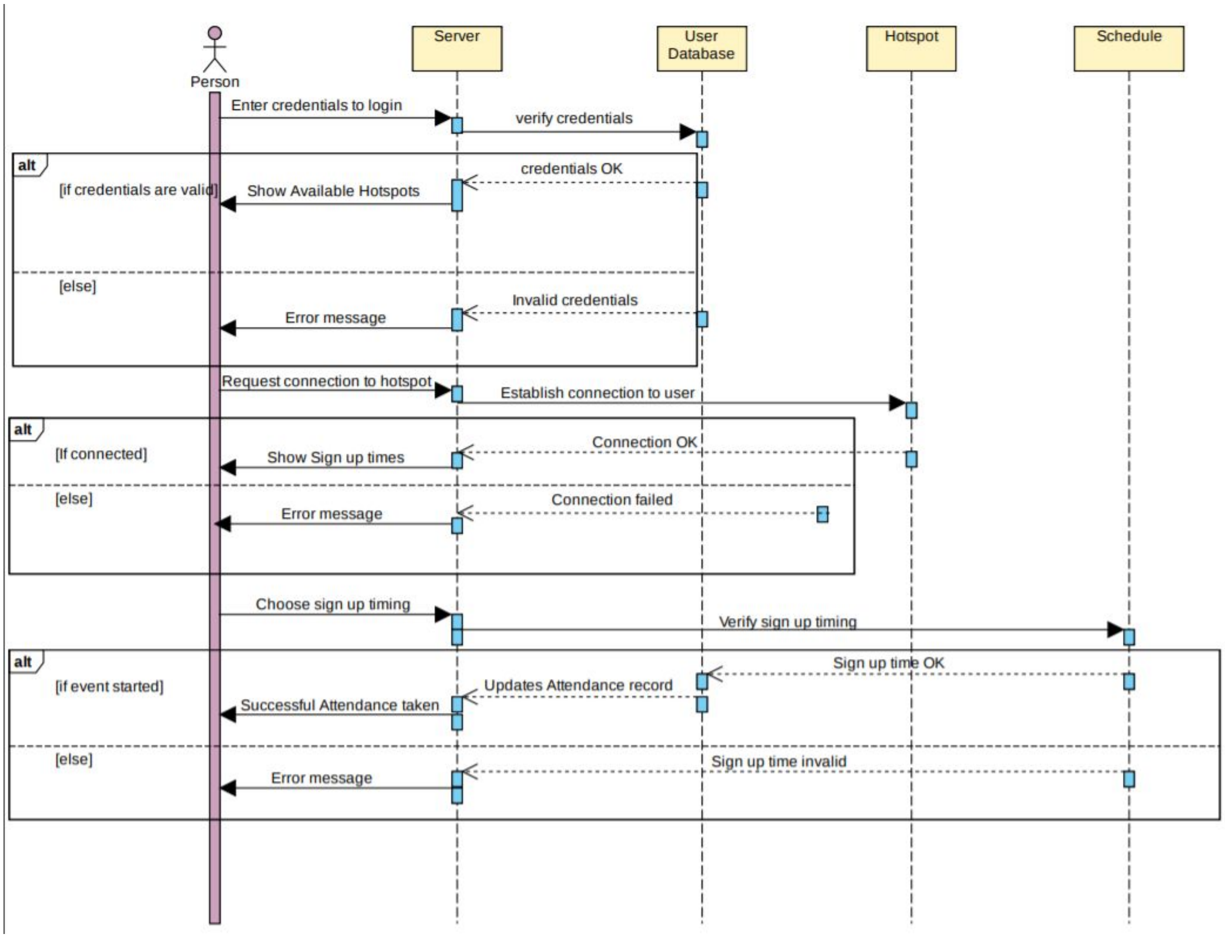
5. Use Case Diagram

Use case diagram

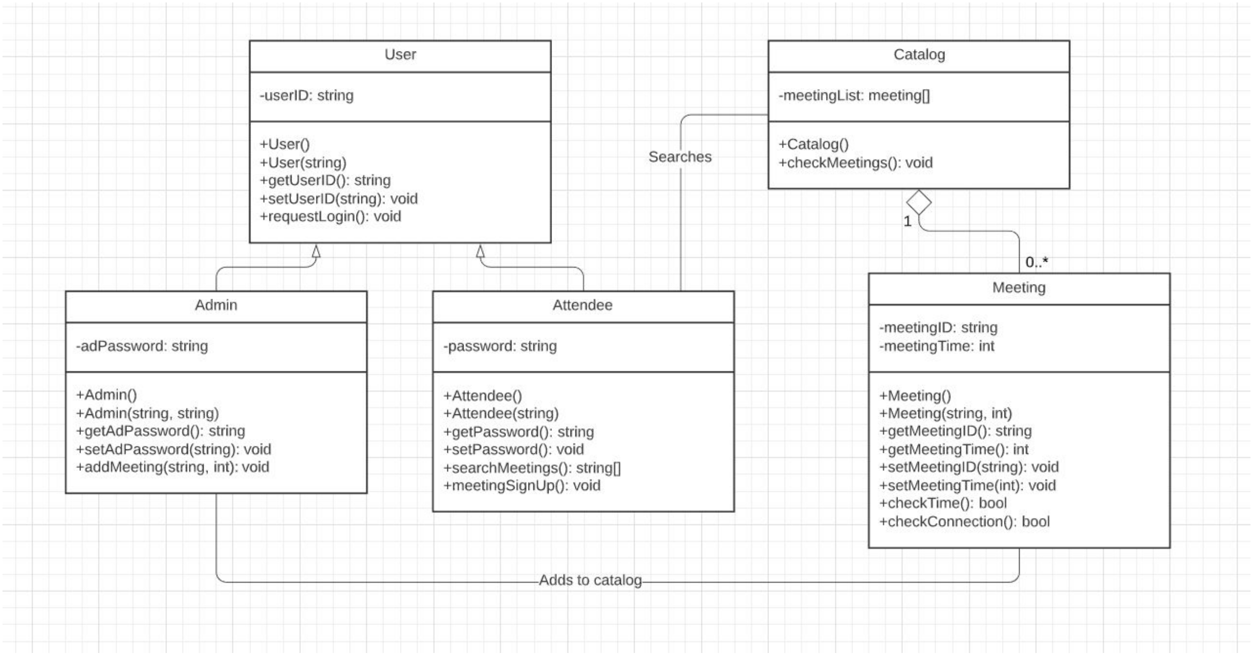
Ethan Villanti | March 13, 2020



6. **Sequence Diagram**



7. **Class Diagram**



8. **Architectural Design**

After discussing with the team, we chose the client server architecture, because it is most suited for an attendance hotspot application. Since the data and user share a single interaction (student being present or absent) the application will not be overly congested and slow down performance ability. Students are also quick to input their answers so traffic congestion should not be an issue at any point. The simplicity of administering a client server architecture will also keep costs and maintenance down. The server can provide the appropriate resources and processing power while clients do the simple task of relaying yes or no messages back. All these positives make the use of a single server and multiple client architecture the best setup to successfully administer an attendance app.