Programming Pangolins

Blake Humes, Brooks Robinson, Ryan Porter, Evan Slominski

CSC 470

Mr. Jenkins

10/23/22

# Must Read

* If any links do not work, contact Ryan Porter to get that corrected
* Warnings
  + This should work on Windows computers (10 or 11)
  + This program will have race conditions, so ensure that 2 users do not do conflicting operations
  + The computer must be in 12 hr. format (not 24 hr. format). This is a bug that is currently being worked on
* (or see training video on how to install)
  + Unzip the folder
  + Hit “Setup”
  + Install from there
* Discord Invite:
  + <https://discord.gg/X76ZCM26uW>
  + In case you want to see the history of our conversations and discussions
* One Drive Invite:
  + <https://dakotastateuniversity-my.sharepoint.com/:f:/g/personal/ryan_porter_trojans_dsu_edu/Es29M6GCSSlOgob5VHuxHJIBFn5xGhGpEYoqTunJn8XDyw>
  + This link should work for your DSU email.
  + This will contain the most up-to-date designs and documents, but those should be included in the zip file. However, the one thing that was too big for the zip file was the recording of our voice-chat meetings. They are in the “Videos” folder.
* GitHub Invite:
  + You should have received an invite on 12/5/22, if not, contact Ryan Porter for another invite
    - Graphical user interface

      Description automatically generated with medium confidence
* Bug Tracker Login
  + You should have received a user invite on 12/5/22, from Mantis DB, if not, contact Ryan Porter for another invite
    - Graphical user interface, application

      Description automatically generated
* Database login (view only)
  + See the database word document in the zip
  + First string is username; second is password
  + Host is web address
* Rest of zip file
  + Updated Design, Requirements, and Planning documents
  + Installation Files
    - Should be in a zip folder
  + This document
  + Training Links
    - <https://www.youtube.com/playlist?list=PL3sVPaIoTs8A-FzFCBjkK6xn_wJbziPFT>
    - Or see OneDrive folder
  + Testing Documents/Refactoring
  + Maintenance Documents
  + Completed Project Plan
  + Summary Link
    - <https://www.youtube.com/playlist?list=PL3sVPaIoTs8A-FzFCBjkK6xn_wJbziPFT>
    - Or see discussion post or OneDrive folder
* Login to the Program
  + Username: 12
  + Password: 4645
* Not Included
  + Meeting video files
* Covering each rubric topic
  + Complexity
    - We used AWS for our database, which 3 of the 4 of us were unfamiliar with, so it was a good learning experience. Additionally, we used outside libraries to build the pay stubs PDFs. We tried to be as thorough as possible with the testing to ensure the program worked in most cases. All of this took some time, and we relied on our past knowledge to help guide us through the project since we had to think about many different things (encryption, databases, classes, etc.).
  + Completeness
    - Yes, the project works and meets the requirements (see acceptance testing).
    - Additionally, Ryan Porter had his boss (a controller) test it, and he said that it works well for a simple payroll program.
  + Summary
    - See: <https://www.youtube.com/playlist?list=PL3sVPaIoTs8A-FzFCBjkK6xn_wJbziPFT>
    - We did a summary video of what our project is, how it works, the code behind, and the wrap-up (what went well, improvements, etc.).
  + Final Peer Review
    - We all turned in our peer reviews.
  + Testing
    - Actual code for test cases can be found on GitHub
    - Can find the testing logs in the zip folder
    - Overall, we did a lot of testing. We unit tested most of our methods individually to ensure that they work as expected with edge cases. Additionally, we had installation tests to test to see if the program would work on other computers and connect to the database. Also, we reviewed our requirements to ensure that they are met with acceptance testing. Lastly, we did brief system testing to make sure that we could add, delete, etc. correctly and that bad input would not make it through.
  + Code Quality
    - We used refactoring (using the JetBrains tool) to help make the code more readable and less redundant.
    - As we programmed and reviewed the code, we used descriptive names for variables, tried to modularize it as much as possible so that the parts are not too dependent on each other, and included comments when necessary to explain what the code should be doing.
  + Intangibles/other
    - We did a lot of extra work that I think helped make the project better:
      * We use refactoring code to make the code more efficient and readable
      * Ryan Porter used Memory Profiling to fix some memory leaks
      * We did a lot of testing and different forms of testing
      * We used LiveShare in VS to help fix some errors