Kevin Huang

ELEC3225

Assignment 3

June 10, 2021

Assignment 3

**Waterfall:**

Requirements Definition:

**# Clear basic requirements of the system - Hank**

* Database of users (be able to hold at least 100 students, 10 instructors, 1 admin)
* Database of courses
  + CRN, course name, time periods, instructor for course
* Three users – student, instructor, admin
  + Student abilities: register classes, see available courses, see schedule
  + Instructor abilities: see available courses, see course roster
  + Admin abilities: see all data, edit courses/users/schedules
* Should be able to handle multiple semesters
* Have the ability to print schedules, scheduling preferences
* Will need to be tested thoroughly

System and Software Design:

Hardware:

-PC or laptop

Software:

**#What is used to build the GUI? - Hank**

* System will be on either C++ or Python language
* System will also include a database from Wentworth
* System will be interacted through the input from user

Implementation and Unit Testing:

* System will have to run and tested thoroughly several times throughout the process to make sure all components work properly
* These components could include the functions of each user and the classes created from the database created through the user input
* System components will be tested using the user interface from either laptop or pc

**#System components could be tested earlier when the GUI has not been built. And once you have a GUI and integrated the function into the GUI, a second test can be conducted.**

* Create code by steps and move onto the next component when one component is working properly
* Once functions of one user is satisfied, move on to the functions of a different user

Integration and System Testing:

* Combine all components of the system and test to see if all work simultaneously
* All components should include the three different user and their functions as well as the databases
* See if the main functions work just as the requirements state
* System must be able to do functions that each user can do

Operation and Maintenance:

* Once system works properly, send the program out for users to test on
* See if there are any issues by listening to feedback of system
* If system does end up having issues resolve issues by reviewing the code and release patch notes to fix any errors
* If new updates are required in the future, make any updates to the system that follows the requirements

**Incremental Development Model:**

Initial Version:

-Initial version should include the main function and the classes for each different user

-Each class will have the different user functions, but will remain empty until later when the functions are added

-System should be able to take user information like ID, and name.

-System should have a menu for user to select from

-This menu will have options for all three users and inside the option will be a sub menu for the users to choose the different functions

-setup of databases for the user information can begin

Intermediate Version:

**#Database can be created and integrated to this version.**

-Add on functions for each user and make sure the functions work properly by testing it using a small test group

-Once functions work properly, place functions into each of the user classes and test to see if there are any issues

-System should work properly with all functions for each user

-Integrate system into a database to see if data from user can be saved in the database properly

Final Version:

-Test system with a proper test group and see if there are any issues with the system and the database

-Make sure data from all three users end up in the database and that all the functions work with the data from the database

-Once system and database work properly together, release system

**Integrate and Configure:**

Requirements Specification:

* Database of users (be able to hold at least 100 students, 10 instructors, 1 admin)
* Database of courses
  + CRN, course name, time periods, instructor for course
* Three users – student, instructor, admin
  + Student abilities: register classes, see available courses, see schedule
  + Instructor abilities: see available courses, see course roster
  + Admin abilities: see all data, edit courses/users/schedules
* Should be able to handle multiple semesters
* Have the ability to print schedules, scheduling preferences.
* Will need to be tested thoroughly

Component Analysis:

-Existing software that could possibly be used

=> <https://github.com/Ri2parna/Course-Registration-System>

=> <https://github.com/ANUHYA1/Online-course-registration>

=> <https://github.com/pockemon/Student-Teacher-Interaction-Portal>

Requirements Modification:

* From Ri2parna code:
  + Could use admin and student functions including add courses, add student, view schedule, etc.
  + One way to configure is to also add feature for the admin to also add different users besides students
* From ANUHYA1 code:
  + Could use multiple functions including the add/drop courses and see schedule
  + Implement functions into the system by adding them into the classes
* From pockemon code:
  + Use database structure and add on to the database to fulfill all requirements for a database
  + Could also use user logins for each different user and to log in the system with ID and name instead of asking for that information.
  + Have a function for the admin to add users so that both students and instructors can log into system through that way

System Design with Reuse:

* Once the modifications are made with the implemented code, work on making sure the system works together properly
* Makes any changes so that the requirements are made and test to see if modifications work

**#The classes and GUI can be reused.**

Development and Integration:

* Combine all components of the system and test to see if all work simultaneously
* All components should include the three different user and their functions as well as the databases
* See if the main functions work just as the requirements state
* System must be able to do functions that each user can do

System Validation:

* Once system works properly, send the program out for users to test on
* See if there are any issues by listening to feedback of system
* If system does end up having issues resolve issues by reviewing the code and release patch notes to fix any errors
* If new updates are required in the future, make any updates to the system that follows the requirements

**Hank was here**