DELIVERABLE 2 and 3– PROJECT PLAN

**Demo Functionality (Developer)**

PLEASE VIEW LECTURE 8 – DISCUSSES ABOUT GUI

**TO-DO-LIST (Developers):**

***Deadline: 13th April 2023***

* ***red text = High priority (must be done ASAP, mandatory for marks)***
* ***blue text = Low priority (optional; may complete after all high priority tasks are completed)***
* ***green highlight = COMPLETED TASK***

***Jacob will be our backup developer – developer team will ask Jacob for guidance if we’re unable to solve a task before deadline!!***

***Aymen = Code functionality / Database connectivity***

***Rati/Ethan = SQL / Design layout***

**02/04/23 – 04/04/23 --- COMPLETED**

* Load the table (Aymen)

StockControl

* Add table, search ticketID, update table, delete table (Aymen)
* cancel button (Aymen)
* Create SaleControl, SaleReport classes – design layout must be identical to StockControl (Rati/Ethan)

**04/04/23 – 06/04/23**

* Modify Refund class so that the design layout is identical to StockControl (Ethan)
* Modifty Alert class – design layout must be identical to StockControl (Ethan)
* Create new Alert **database** – layout identical to “GUI Design.pptx” - slide 10 (Rati)
* Remove “airlineID” & “managerName” and replace it with “agentID” on **Ticket** database (Rati/Aymen)
* Modify CustomerManagement - design layout is identical to StockControl (Rati)
* Modify AddCustomer - labels, textfields is identical to “Customer” database (Rati)
* Enable AUTO-INCREMENT for ID (Rati)
* Copy database functionality from StockControl to: CustomerManagement, SaleControl, SaleReport, Alerts, Refund (Aymen)
  + Remove all ID fields on all database classes (Aymen)
  + Add combo box – blankType, paymentType, customerType, etc (Aymen)
* Add functionality to Backup/Restore & Print – exported as **pdf** (Aymen/Rati/Ethan) .
* Make another Login page when the user clicks on SaleReport – this will be different to the other login page as this will be Admin based; can’t create new accounts; only one login detail; saved on a different text file (Rati/Aymen)
* Login – passwordTextField will be anonymous - unless its ticked by “show password” (Aymen)
* WelcomePage – outputs “Welcome [username] (Aymen)
* Add JCalendar / JDateChoose onto Database (Aymen/Rati/Ethan)

**06/04/23 – 10/04/23**

* Add new columns in “Travel\_Agent” and “Office\_Manager” databases called **username** and **password** (Rati)
* Make a **new Database table** called “Admin” and make it identical to “Agent” database format – including username and password (Rati)
* Create multiple Welcome pages depending on different role (Aymen/Rati)
* Create EditAgent, EditManager, EditAdmin class – design layout must be identical to StockControl (Rati)
* Copy database functionality from StockControl to: EditAgent, EditManager, EditAdmin (Aymen)
* Add JavaDoc for all classes we worked on (Aymen/Rati/Ethan)
* Complete any missing tasks ASAP (Aymen/Rati/Ethan)
* Final code must be completed (Aymen/Rati/Ethan)
* Let the testers know that it is completed! (Aymen/Rati/Ethan)
* Make design layout look professional & attractive (Aymen)

**10/04/23 – 12/04/23**

* PREPARE FOR THE DEMO PRESENTATION (Everyone)
* Read the demo presentation PowerPoint which will be completed by Monday.
* Read the scripts that are made for everyone (The people who have specialised in an area of development will only speak about their area of development).
* The managers will mainly be engaging with the client in terms of non-technical questions and promotion of our company and services (Since Air Via will renew their contract with us to carry out maintenance for their software).
* **Everyone must attend** as explicitly stated by Martin on a few occasions in the lecture. Failure to do so without good reason may lead to a heavily reduced grade or failure of the whole module (This module is equal to 60 credits which is two normal modules, so failure would mean failing two modules at the same time).
* Leading on from the last point; **attending doesn’t necessarily mean you have to do some talking,** the only thing that is required of you is to be present and at least act like your engaged in the presentation. In addition to this it would be very nice to see some of the people who have not been able to meet the rest of the group in person but still contributed to the team project.

**GUI Forms:**

* Login
* Sign-up
* Change Password
* Welcome
* **Customer Management** (DATABASE: Customer)
  + Only admin roles (Manager, Admin, Advisor) can: *(lecture 8 – page 32)*
    - Edit customer details
    - View general reporting of customers
    - Add discounts for valued customers
* Add User
* **Stock Control** (DATABASE: Ticket) *(lecture 8 – page 29)* 
  + Adding and removing **blanks** from a **travel agent** (company)
  + Searching for **blanks**
  + Stock Turnover **report**
  + Assigning, removing, and reassigning **blanks** to a **travel advisor**
* **Sales Control** (DATABASE: Air\_Ticket\_Sale) *(lecture 8 – page 30)*
  + Making a **sale** as an **advisor**
    - Different **customer types**
    - Different **sale types** (interline and domestic)
    - Different **payment types**
  + Recording **payments**
  + Refunding **tickets**
* **Sales Reports** (DATABASE: Air\_Ticket\_Sales\_Report) *(lecture 8 – page 31)*
  + For or by an **advisor**
  + For the whole **agency**
  + Consider the different **reports** and calculations required etc.
  + Consider who should be able to produce the **reports**
* **Alerts** (DATABASE: ???) *(lecture 8 – page 33)*
* Refund (DATABASE: Payment)

**Implementation Report (Tester)**

*PLEASE VIEW LECTURE 5 – IT DISCUSSES ABOUT IMPLEMENTATION REPORT IN MORE DETAILS*

*PLEASE VIEW OOAD LECTURE 9-IT GOES INTO DETAIL ABOUT HOW TO IMPLEMENT THE TESTS*

Testers’ Task:

* Abbib -> Writing implementation report, testing plan, creating presentation,testing four use cases and deployment diagram.
* Uzair ->Testing three use cases
* Ishmail ->Testing three use cases
* Jacob ->Carrying out unit testing

Marking: *(lecture 5, page 45)*

Implementation Report [20 marks]

* Software architecture/ Compilation/ Run-time components [8 marks]
* Testing plans and reports [12 marks]

Structure: *(lecture 5, page 42)*

* Needed for product maintenance
* Failure to supply this will result in ZERO marks being awarded for the whole of the Implementation!
* Must include the following:
  1. Compilation *(lecture 5, page 47)*
     + What are the source files, programming language and compiler used (version, compiling and linking options, script files, e.g. main files). If applicable, provide a project file description (forms, data modules, etc.)
     + A list of created binary files (.exe + .dll or .jar files for Java):
       - their short description and where they should be located at run time (directories on the target machine). • If more than one .exe file is created (e.g. main file + backup/restore utility), a clear description should be given of who, and how, is going to use them.
  2. Appendix *(lecture 5, page 47)*
     + Commented source code files (in a zipped file for the online submission)
  3. Run time – Testing *(lecture 5, page 48)*
     + As a minimum, a list of the COTS (commercial off the shelf) software components that must be provided (static libraries, DLLs, etc.) which the application needs to run properly.
     + Dependencies between components (DLLs, etc.) to be shown, e.g. using UML component diagrams.
     + Deployment of the run time files on the target computer(s)
  4. A testing plan for (as defined above) and testing results *(lecture 5, page 48)*
  5. Documentation *(lecture 5, page 48)*
     + Technical documentation of your codebase and database as required
     + User documentation to aid new users to learn the system

Things to do: *(lecture 5, page 43, 51-56)*

* Define the programming language and develop the codebase
* Develop the database in MySQL
* Test and refine
* Optional: Add UML Component / Deployment Diagram
* Present and demonstrate your final software product

Preparation: *(lecture 5, page 44)*

* This deliverable must be a single, well formatted and presented report
* Insert your diagrams into the document to preserve your page numbering etc.
* Ensure all diagrams are an appropriate size (readable)
* You may decide to use UML Component (or even Deployment) diagrams in the report, but this is not mandatory. Textual description is also acceptable, provided it is complete and correct

**Weekly Plan / Gantt Chart**

**Developers:**

*(DUE Thursday, April 13th)*

* Week 9 (20/03/23 – 26/03/23):
  + Each member will work on their chosen GUI form
  + Due to us having so many coursework, it’s best to start early from now and do at least 30 minutes DAILY on our task.
  + Functionality is main priority. (Example: Making sure buttons works, adding text fields, etc.)
  + Any changes that you make on the code – please push it onto GitHub
* Week 10 (27/03/23 – 02/04/23):
  + Databases (Example: Creating a new account will produce a database of emails and password, reports, etc.)
  + JavaDoc (discussed in lecture 8)
  + Will be discussed further soon on what we should do
* Week 11 (03/04/23 – 09/04/23):
  + Final code must be completed
  + Make the GUI look appealing and professional!
  + Let the tester know that the program is completed
  + Submit the coursework!

**Testers:**

*(DUE Sunday, April 16th)*

* Week 9 (27/03/2023-02/04/2023):
* Set up the Implementation report.
* Revising OOAD lecture 9 for testing.
* Writing testing plan
* Revising OOAD lecture 8 for UML deployment diagram.
* Maybe writing the user documentation since we already know what the system should have?
* Week 10 (03/04/2023-09/04/2023):
* Started testing our allocated packages as soon as they are implemented.
* Publishing the test results
* Finishing the technical documentation for our implementation report
* Finishing the deployment diagram
* Week 11(10/04/2023-16/04/2023):
* Inputting the compilation details into the implementation report
* Inputting run time dependency details and deployment diagram into the implementation report
* Inputting the commented source code into the submission
* Checking over the implementation report
* Submitting the implementation report!

|  |  |  |  |
| --- | --- | --- | --- |
|  | Week 9 | Week 10 | Week 11 |
| GUI |  |  |  |
| Code implementation |  |  |  |
| Database |  |  |  |
| Testing plan |  |  |  |
| Unit/use base testing |  |  |  |
| Implementation report document |  |  |  |
| Submit deliverable 2 |  |  |  |