Criterion E

Meeting the success criterion:

The client's feedback demonstrated great enthusiasm and evidence(See Appendix 4) that this program fulfills all the specified success criteria. Additionally, this program closely resembles the originally proposed GUI designs that were met with enthusiasm and approval. However, the client did propose some recommendations, one of them being it would be great to have names rather than employee numbers in the generated PDF, as she now prints the PDF and hangs it in the shop she believes that adding names adds a personal touch, rather than simply displaying their numbers. This however, is mainly UI related, it would not take much to translate the employee numbers into names, using the recursive *returnEmployeeFromList()* method.

As seen in the following table: each element of the success criteria has been tested in relation with the test plan in criterion B. These tests have all been fulfilled to an appropriate level as specified by the client.

No	Action to test	Method of testing	Success criteria	Tested	Fulfilled(Yes or No)	Comments
1	Login	Enter correct details and login screen is displayed	1	✓	Yes	Works well, and can handle errors.
2	Add employees	After employee's information is entered, the employee is added to the DB and Employee List	2	✓	Yes	Works well. Employees can be added.
3	Delete employee	Employee and his information are deleted and un-serialized when deleted.	2	✓	Yes	Works well. Employees can be deleted, which un-serializes them.
4	Edit Employee	The employee's information can be modified after its original addition	4	✓	Yes	Works well. Employees can be edited after original addition.

5	Shift generation	The shift schedule is generated according to user's information, and displayed appropriately	5	✓	Yes	Works well, although it could be extended in the future, by categorizing employees based on the shop they are in.
6	PDF for employee's data	Generate PDF based on all of the employee's information in the correct folder	6	✓	Yes	Work well. Could add the date it was generated for future development
7	PDF for shifts	Generate PDF based on the shift generated, in the correct folder	7	✓	Yes	Works well. Again, could categorize the shifts based on the shop.
8	Salaries and tax bracket adjustments for pay slip pdf generation	Pay slip is generated according to the user's input tax brackets, deductions are correct, and the pdf is generated in the right folder accordingly	8	✓	Yes	Works well. However, an issue occurs when choosing the Indian Rupee, whereby the character is replaced by its Unicode
9	Serialization of data	Data is generated accordingly in a .ser format file. Data can be read without corruption of the stream header, and data can be removed when requested to be removed, without impacting the order of the data. DB serialization also works without any issues.	9	✓	Yes	Works well, and allows two different methods of implementing features, as employee data is available from both source, both with advantages and disadvantages.
10	Search for Employee	Program can return an employee based on data given by user. The program gives a list of the best fit employees for data given, such as name, last name.	3	✓	Yes	Works well. Could add more parameters in future development.

Recommendations for future improvements

The client recommended that a button be added on the generate shift UI, that once pressed would open up a new email(See Appendix 4), and the generated shift as a .PDF attachement, which could be implemented quite easily using the java AWT’s *desktop* class that already has this functionality.

As suggested by the client(See Appendix 4), the system can handle employees from multiple shops, but the shift generator lacks a function to generate shifts for a specific shop, forcing users to exclude employees manually. Although technically functional, this is not user-friendly. To address this, we can leverage the existing "exclude employee" arrays in the generateShift() method parameters. By populating the array based on the user's shop selection, we can automatically exclude all employees not in that shop, and display only the relevant employees in the JTable.

Moreover, as mentioned by the client in Appendix 4, different languages would be beneficial to the overall functioning of the program, as some employees only speak French. This could be done easily using Java Internationalization (i18n) API, which provides a framework for developing multilingual applications. This API allows to externalize the program's text and provide translations for different languages in resource bundles.

Word count: 349