DSB001: Final evaluation exercise

Purpose

The purpose of this document is to specify an online project that the students on the Skills Bootcamp in Software Engineering will complete as part of their initial training with Aston University.

1 Introduction

1.1 Guidelines

Please read this document in full before you start any work and make sure you understand what is required.

The following general instructions apply:

Time:	You have until 12 January 2022 to work on the given task. Try to complete as much of the project as possible in the time available rather than aiming to finish everything.
References:	You can consult your notes, seek online help, copy and edit previous code as needed. However, we expect the work you submit to be your own.
Questions:	You can ask the module staff for clarification of any issues and for specific technical help. You can book "client" meetings with the module staff (acting as the project client) at any time from 4 Jan until 12 January. The "client" will also be available on Thursday 23 December in the afternoon only. All meetings must be arranged by email at least a day in advance.
Backup:	Make a copy of your work each night to a USB drive or similar!

1.2 Teams

Individuals will be required to work in **teams of two** for the purposes of this exercise. You may choose who to work with but you should give your team a name and confirm your team name and members in an email to `a.j.beaumont@aston.ac.uk`. You can contact other students on Slack if you are looking for team mates. If there is an odd number of students, one team will comprise of three members. Each team will be required to demonstrate their solution and demonstrations will be held on Thursday 13 January and Friday 14 January 2022, (or in the following week if not convenient).

1.3 Schedule

The work schedule for the activities described in this document is as follows (the number of days is only a guide, don't be afraid to use more or less time according to your needs).

Phase 1 (Scoping and Organising, 1 Day)

- The module facilitators will deliver a briefing about the exercise.
- Teams will be given time to prepare for a 'client meeting'.
- The 'client meeting' will take place on 4 January 2022 is a chance to present material to explain how your team plans on delivering the project and organising the work.
- Teams will have an opportunity to ask the client for clarification regarding aspects of the requirements.
- **Milestone**: You team should complete and demonstrate the first **M** requirement in a short presentation at the end of the day.

Phase 2 (Development and Testing), 3-4 Days

- Teams will hold a Scrum meeting (or similar; no more than 15 minutes) twice a day, during which team members will report on previous progress and will be given new tasks.
- Teams will also have access to the client for clarifications.
- Teams will be expected to demonstrate their progress to the client.
- **Milestone for day 2**: Your team should complete and demonstrate the next three **M** requirements in a short presentation at the end of the day.
- **Milestone for day 3**: Your team should complete and demonstrate the final **M** requirements and first two **S** requirements in a short presentation at the end of the day.

Phase 3 (Final Development, Testing, and Demonstration Preparation, 1-2 Days)

- Twice-daily scrum meetings and access to client will continue.
- Teams are expected to balance work between completion of tasks, checking of deliverables (code, release notes, etc.) and preparation for the final presentation.
- **Milestone**: Your team should complete and demonstrate as many of the remaining MoSCoW requirements as you manage to complete in a short presentation at the end of the day.

Phase 4 (Demonstration and Retrospective)

- The final day should be used for finishing touches and final preparation for the demonstration.
- Solutions shall be handed in by Midnight on Wed 12 January.
- A complete submission must contain all of the below:
  - A zip archive with the project source code in full (all Java and SQL source files along with any other code necessary to run the application).
  - The presentation slides. If your class diagram, ERD, or component diagram are not included in the slides then they must be included separately in your submission.
  - A short user guide explaining how to install/set up and use the application. Any known bugs should also be documented and explained/justified here.
  - A reflection document including answers to the questions suggested in section 3.3 below, as well as any other considerations regarding your teamwork experience.
- **No late submissions are allowed**.
  - If any of the items described above is missing, or submitted late, the project will not be accepted.
  - If any source files are missing, thus preventing the assessors from running the project and replicating the claimed results, the project will not be accepted.

  Submit something that is working even if it does not fulfil all of the requirements!

- In the final presentation and demonstration, each team will be allocated time to present and then additional time for questions. See section 3.4 for more details.

2 Project Scope and Requirements

The project is to create a new booking system for The Theatre Royal. The User Interface will be text-based using the console. The can be a separate application for Theatre Employees and for the General Public to use to book tickets.

2.1 Outline

The theatre wants to offer information regarding forthcoming shows, and to allow people to purchase performance tickets online.

The interface should ensure that users of the application to find the following information for any show:

- The title of the show;
- The type of show (theatre, musical, opera, concert);
- A description;
- When performances of the show are staged (times – matinee or evening – and dates);
- The duration of a performance;
- The price of tickets for a performance;
- The availability of tickets for a performance.

A show may be staged multiple times, sometimes twice in one day (matinee and evening performances). A show will be of a given type, as described. Theatre shows, musicals, and operas may be presented in languages other than English; concerts do not have a language. Musicals and operas may or may not have live musical accompaniment. Concerts always include live music. Where live music is a part of a show, the name(s) of the principle performers or performing group should be displayed prominently within the details of the show.

The theatre has 200 seats, all of which are available for any performance. These are split between the stalls (120 seats) and the circle (80 seats); currently the theatre charges the same amount for seats in the stalls and seats in the circle for a performance, though this may change in the future. It is important that seats are not double-booked for a performance. The price of a ticket varies by show. Concessionary tickets (children under 16, students etc.) are priced at a 25% discount.

Users should be able to browse details of shows freely, though they must provide their name and address and credit card number if they are to complete an order for tickets.

It is important that the new application is available for user acceptance testing by the deadline (Midnight/Wed 12 January).

Alongside the application functionality, the Theatre Royal have also commissioned a study regarding attitudes to theatre and theatre attendance in the UK. They are interested in your advice, based upon your analysis of available data, regarding ways in which they can use the new application (particularly if it were converted into a Web Application) to increase interest and attendance at the theatre. The theatre has suggested the following sources, but you are free to find and use your own sources if you are aware of something more appropriate or useful:

- https://yougov.co.uk/opi/browse/Going_to_the_theatre
  A demographic overview, rather than a set of in-depth statistics.
- https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/592701/Disability_and_theatre_table.xlsx
  Theatre engagement statistics regarding a particular societal group.
- http://www.thecreativeindustries.co.uk/industries/arts-culture/arts-culture-facts-and-figures
  Many different reports on various aspects of the arts, produced by the Creative Industries Council.
- http://www.artscouncil.org.uk/sites/default/files/download-file/Analysis%20of%20Theatre%20in%20England%20-%20Final%20Report.pdf
  Analysis of Theatre in England by the Arts Council: some statistical data included both within the report and also in the technical annexes.

You are not expected to make recommendations upon the types of shows presented, pricing etc. but rather upon the ways in which an application or a website would help to support and enhance the business. Your final presentation should contain concrete suggestions regarding the application/website and the theatre's use of it. **Make sure to appropriately acknowledge all data sources used in your report**.

2.2 High level tasks

The site will let users browse shows/performances, search for specific shows or dates, select a number of tickets for show(s) for purchase, and to complete their purchase of those tickets. The back-end model should support all of these actions, along with any others arising from them, or from the requirements.

You will need to:

- Design and implement the underlying database:
  - Determine the entities and attributes within the requirements, along with any other data necessary, and identify that data which needs to be stored within the database.
  - Design and implement the appropriate, normalized, table structures to manage that data.
  - Design sample data to allow the structure and application to be tested and demonstrated.
- Design and implement the back-end model for the shows/performances:
  - Discuss the design of the class diagram in your groups based on the project specification (noun-verb analysis).
  - Discuss the relationships between the classes in your diagram (aggregation vs composition).
  - Analyse the way the back-end model will connect to the front end interface (how to integrate the JDBC API in your implementation).
  - Determine how to connect this middleware to the front-end interface.

  - Implement all classes within your model (start with class fields and then decide on class methods).

- Implement the front-end interface:
  - Users should be able to browse shows/performances.
  - Users should be able to search for a particular show, or view performances on a particular date.
  - Users should be able to select tickets for purchase. Users may purchase multiple tickets, possibly for multiple shows, in one transaction.
  - Screen(s) must exist to allow the user to see which tickets, for which shows, they have selected for purchase, and to allow the user to enter the details necessary to complete a purchase.
- Produce your data analysis report regarding the website and the Theatre Royal's use of it.
  - Analyse several sources of reliable and relevant data (use those suggested by the theatre as well as finding others).
  - Look for relevant trends or correlations with the data and formulate a hypothesis based on that.
  - Test your hypothesis by interviewing other apprentices: consider the ethics and purpose of your questioning. Collect their feedback and see if your theory is confirmed or proven wrong.
  - Use your conclusions to suggest a list of concrete modifications in the algorithms, data presentation etc. implemented in the new site with the end goal of improving the site.

2.3 Product features

You are expected to complete all the 'Musts' in the backlog. Note that a good quality back-end model is crucial for the efficient implementation of product features. If time permits, the solution should be extended to maintain the other features within the system. These features are detailed in the backlog:

What is MoSCoW? *http://en.wikipedia.org/wiki/MoSCoW_Method*

Feature/Story	MoSCoW
Browse shows and search for a show or a specific date	M
Display search results	M
Select tickets for purchase, adding them to a 'basket'	M
Place order screen showing order total and allowing users to place an order. Purchase screen requesting credit card details and allowing users to make the payment.	M
Produce a report with suggestions for improvements based on third-party data and reports.	M
When an order is placed, the user can choose to have tickets posted to them, or to collect their tickets at the box office. Posting tickets costs £1 per ticket, unless at least one concessionary ticket is purchased, in which case the cost for posting is £1 in total. If all tickets being purchased are concessionary, the theatre will offer to post them for free.	S
The option to have tickets posted will only be available if all of the performance(s) for which tickets are purchased are at least seven days in the future.	S
Users are able to perform a new show/date search from the screen displaying the result of a previous search.	S
System administrators are able to add new shows and performances to the site.	C
When an order is placed, a file containing purchase details is available for printing.	C

3 Guidelines and Hints

3.1 General Instructions

All the code should adhere to the guidelines for good practice provided. For example:

- Use software patterns and remember the core object oriented concepts you have learnt.
- Follow the standard naming conventions.
- Use clear and well-structured code.
- Use appropriate comments.
- Give meaningful names to variables.
- Have unit tests.

ImplementationPatterns.pdf contains a list of good coding practices and is available on Blackboard. This should be referenced and followed accordingly.

The finished screens should have a professional “look and feel”. It is your job to make sure that the screens present all the required data in an understandable and professional layout with appropriate help and messages as needed.

It is your responsibility to decide, within your teams, on the appropriate information to use for the database tables.

3.2 Development advice

Can you write a one-page document describing the scope of what you are trying to achieve? What are you doing (and equally important, not doing)?

How are you going to break the work down?

What questions do you need to ask?

Who is doing what?

Do not ‘gold-plate’, invent requirements or extend the scope beyond the existing specification.

How will you know when you are done?

Rewrite the requirements as stories. Convert the one line requirements into something clear with a goal you can test.

Your own implementation will be based on your own unique approach to the problem space. Pay attention to the time constraints in which you are working.

Think how you will communicate in the team. Professional solutions for agile development - such as Trello - are preferred over paper task lists. Code versioning tools - such as SVN or git - are better than sharing code over Google Drive or Dropbox. **Note**: While Trello is straightforward to use, git and SVN require some specific knowledge in order to use them to their full potential.

Think how you will communicate to the client at the end. As you design your solution, think how you are going to justify those decisions in the final presentation.

3.3 Reflection and evaluation

You should complete this element as a team on the last day, by holding a retrospective, and include the answers to the questions below in your presentation. Please also include this electronically in your submission.

- Which bits of the exercise did you find easy/hard and why?
- What went well? Why do you think that was?
- What went badly? Why do you think that was?
- What would you do differently next time, in terms of process?
- Given more time what would you do to improve your solution?
- What skills did you develop or improve during the project?

3.4 Presentation instructions

Each team will have a total of 30minutes to present their work. There will be an additional 10 minute slot allocated to Q&A. The presentation slot will be divided between two talks:

- A **presentation aimed at the client** will be delivered over the first 15 minutes. This should focus on the benefits of your solution to the client, and will describe the given task and your solution. Explain the project goals and demonstrate/justify the implemented features. The high level artefacts (class diagram, ERD and component diagram) may be included in the client presentation only if they are crucial to the narrative (e.g., used to explain solution design and product features). In that case, they should be explained in plain language, without excessive technical detail.
- A **presentation aimed at your managers** will be delivered over the remaining 15 minutes. This should include your high-level diagrams, demonstration of good working and coding practices (e.g. TDD, SQL procedures executed via `CallableStatement` instances to protect against SQL injection, appropriate exception handling, resource management etc.) and any other technical detail that you deem necessary to demonstrate to your managers that you are professional developers/data analysts.

There will be time at the end of the two presentations for Q&A, though you may also be asked questions during the presentation. All team members are expected to talk for equal amounts of time during the two presentations.

3.5 Interacting with the client

The module tutors will serve as technical consultants throughout project week. A client will be brought in to answer questions about project requirements and end features (client meeting timings are to be arranged by email at least a day in advance) as well as to view demos from all teams for each of the daily milestones.

Take a professional approach when interacting with the client. Address him/her respectfully, ask confident and informed questions only after you have carefully studied the specifications (section 2) within your team and make sure to demonstrate how client feedback is integrated in your code from one prototype to the next. Take meeting minutes and share them with the client to get his/her approval. You can refer to the minutes when unsure about the way a feature should be implemented. Also, in the final presentation, you may bring up the client

approved minutes if asked to justify a design decision or to explain the presence/absence of a given feature.

4 Marking

Task		Marks
System design		25
	Java class diagram, ERD, component diagram	
Technical execution		50
	MoSCoW requirements, cohesion and low coupling, testing and exception handling, good coding practices	
Presentation		25
	Talk clarity and conciseness, slide structure and content	
Total		100

The project work submitted by each team will be assessed according to the marking scheme above, resulting in one mark for every team. If every team member contributed equally, individual marks will be the same as the team mark. If contributions were different (according to the fourth piece of the submission pack - see section 1.3, day 5), individual marks will be scaled accordingly. Please note that the maximum individual mark is the team mark (for example, in a team of 3, those who contribute 33% or higher will get the team mark, whereas those who contribute less than 33% will have their individual marks scaled down).