Web Services Assignment

Jamie Prevedoros 11987839

Harry Smith 12000624

Andrew Belekas 99105328

Abstract

This report aims to break down the key concepts and strategies used to develop our Flight Club website. We analyze the methods used to create the functionality of web services to understand how our potential customers can register with our airline company, as well as search and book return flights from city to city. This is done using using a combination of languages and coding techniques identified below

This involves a REST web services which allows users to fetch a list of flights in XML, and a SOAP web service which allows users to login, logout, create listings, make a bookings, view a list of flights and close a listing.

Using trello board, we broke down the assignment specification to better understand what is required in terms of functionality. Each stage was broken down into one of the following categories; Global items, users (viewer, customer, admin) jsp pages, xml and java objects. Please refer to the table below where we break down the functionality of each project component.

Web Pages

Pages	**Description**
Index.html	Redirects the user to Main.jsp
menu.jsp	Contains the navigation to all pages as well as dynamic fields based on the session type (Viewer, Customer, Admin). Allows for a full spectrum of navigation and is present on every page.
main.jsp	Contains the search form of flights, allowing to see flights based on the search in Results.jsp, with a different result in actions allowed based on whether the user is logged in or not.
results.jsp	Contains the list of resulting flights based on the search made previously. Allows the user/admin to either book/view a flight or allows a viewer to login/register before returning and book/view a flight without repeating the search process. Links available on Results.jsp

	allowing for the booking to be made by the user
viewFlight.jsp	This page is responsible for fetching flights by id (findByID). This way users can view their booking details.
makeBooking.jsp	Allows registered users to make a booking by adding the user’s booking request from flights.xml to the user’s account.
booking.jsp	Customers may view, make and cancel their bookings from this page.
cancelBooking.jsp	Terminates user booking. The flight seat becomes available again and the user may book another flight.
register.jsp	Accessed via the menu from any page. Contains a registration form which allows a user to register using the required fields. The form has set parameters for each field, requiring the user enter a valid name, email, password and date of birth. It also checks that the user has not previously registered with the

	specified email. Once the user has entered and submitted valid details, the user details are written to users.xml and the user is returned to the page they came from.
login.jsp	Makes reference to LoginAction.jsp based on the user input in the form present on Login.jsp.
loginAction.jsp	Uses userApplication.java to verify login data and initiate login session. If login is successful user is taken to main.jsp.
logout.jsp	Accessed via the logout button in the menu on any page; Invalidates the user's session and takes them back to Main page after 2 seconds
welcome.jsp	Was used to handle the request for register, display the user's info before refreshing after 5 seconds and returning them to the page they were previously on; functionality was moved to within register.jsp. Page exists but is no longer used.

admin.jsp	Only accessible by a user who isAdmin boolean registers true.

Source packages

Packages	**Description**
Flightseat.java	Returns row, seat number and seat status (available/unavailable), and the user who booked the seat (bookedBy)
Customer.java	Get methods returning name, email, dob
Booking.java	Used to retrieve flight details which include flightID, row, and seatNumber.
Flightseats.java	Get methods to fetch available seats, number of available seats and cancel booking
userApplication.java	Handles the unmarshalling of users.xml
Flight.java	Get and Set methods for the Elements and attributes of a flight; including ID, departure location, destination, an array for seats of each class on the

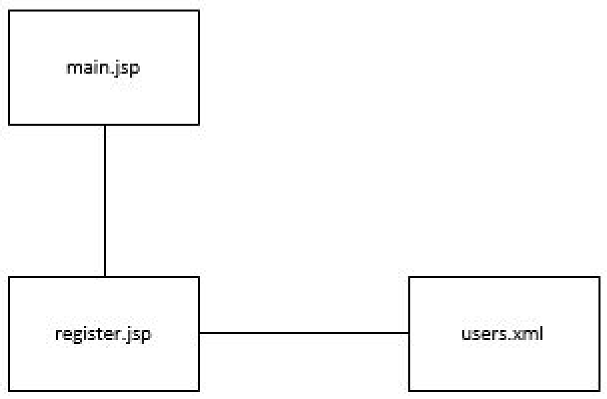
	flight, the price of each class seat in a flight the number of seats available, the number of customers booked on the flight, the date of departure and the date of return.
Flights.java	Handles the list of flights and contains functionality for searching through the flight list and returning flights based on the field information input by the user in results.jsp
FlightApplication.java	Handles the marshalling and unmarshalling of flights.xml
User.java	Get and set methods for the Elements and attributes of a user; including name, email, password, date of birth and whether or not a user is an admin. Also stores a user’s particular Booking.
Users.java	Handles the list of users, allows the creation and deletion of users and is used in the login process, checking the list of users for one matching the login details provided. Also contains all the methods to check the validity of fields

	during registration based on regex logic.
CustomerResults.java	The rest Java Class that is called when a QueryParam for username is given in FlightService.java
FlightResults.java	The rest Java Class that is called when a QueryParam for numofflights is given in FlightService.java
SeatResults.java	The rest Java Class that is called when a QueryParam for numofseats is given in FlightService.java
UsersPrinter.java (flightclub.dom)	This class is used to print users from users.xml. This is not used in the live solution however it was used as a fast and efficient way to view XML files in the server log while testing & debugging the data structure.
flightService.java (flightclub.rest)	This is used to get userApplication and flightApplication for the rest web service

XML files	Description
flights.xml*	Contains entries holding flight details. These include flightID, departure, destination, flight type, flight price,, available seats,, departure date, return date, flight seats, row, seat number, booking status.
users.xml	Contains entries holding user details. These include the user’s email, name, password, date of birth, and account type (customer or admin)

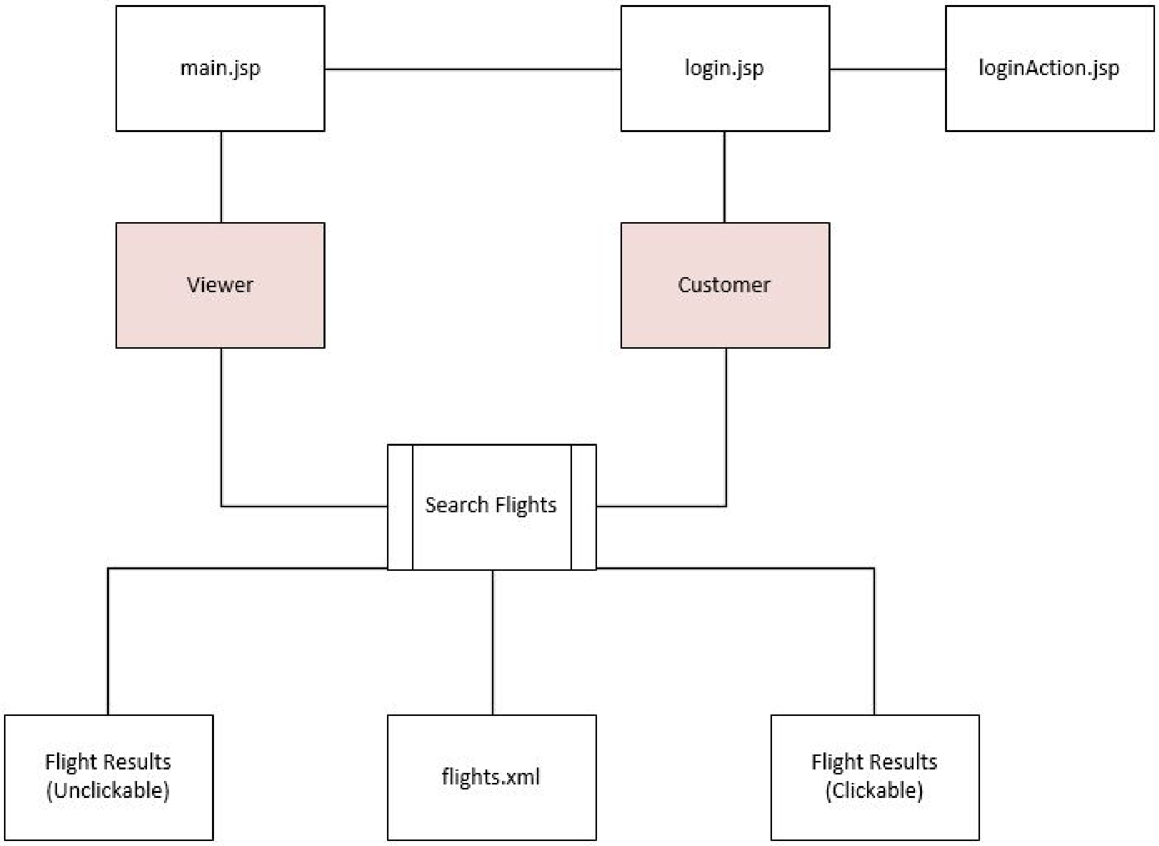
*For flights.xml, we differentiate between economy and business by using <typeofflighte> (Economy) and <typeofflightb> (Business).

Register



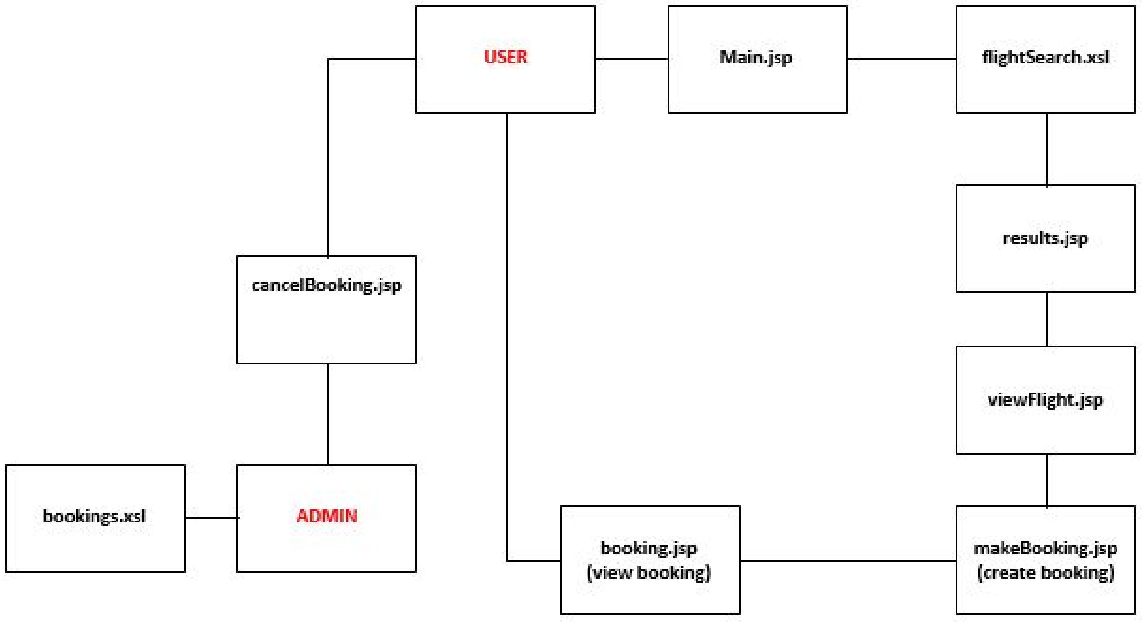
Customers can create their account by navigating to register.jsp from the main page. This is necessary if they wish to book a flight on booking.jsp. Upon registration, they will enter their name, email address and date of birth. These details are then added to the users.xml file. Users are then able to log in with these details. When the user attempts to login, the loginAction.jsp page refers to the users.xml file to validate their details and initiate the login session.

Search Flights (Results page)



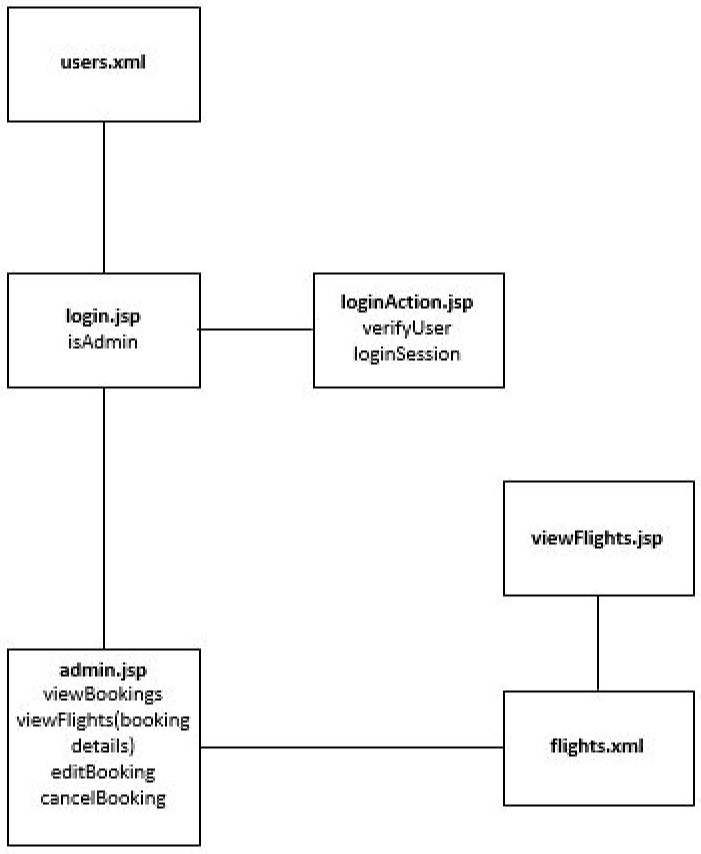
Customers are able to search available flights by entering their flight departure and return dates, origin and destination cities. The SOAP web service function has been implemented to retrieve and list available flights from the flight.xml file. The diagram above shows how the flight information is retrieved upon search. Depending on who initiates the search, a viewer is only be able to view the results, whereas a registered user is able to select the flight and make a booking.

Make booking



In an attempt to understand the logical structure of how users are able to book and view their flight, we have created a viewFlight.jsp page which lists the user's current flight by fetching the flightID. This page views the user's current flight details including flight seat (row/letter), origin/destination, type (economy/business), departure and return dates, price and booking status. From here the customer will have the option to book a flight, edit or cancel their current booking. Admin users have the ability to view customer flights, as well as cancel customer bookings.

Admin



In order to verify the user type we have implemented the isAdmin attribute in the users.xml file. Users with the admin attribute are able to login and view all customer bookings and are given edit/cancel functionality. This is a necessary distinction we have implemented to meet the assignment specification, granting admin users such as airline staff access to customer bookings.Users without the isAdmin attribute are simply customers with minimum booking privileges.

Reflection on experiences

James

I felt during the project development that i was good at determining the logic required to complete the tasks I was assigned, however I struggled greatly with syntax and method/class usage. In setting out a skeleton framework for the assignment, I utilised this to create what I believed to be a fairly accurate set of comments in the initial files creates which broke down the logic flow of how the project will work and where the interactions between classes would occur. Upon discussion with both my group members and tutor, some of this logic changed throughout the course of the project with my improved understanding of the task at hand and the processes required to complete it. I feel this was the most valuable activity I undertook, specific to the project, as it improved my knowledge across almost all areas of the project.

Further, I faced issues in relation to my lack of pre-existing knowledge around GitHub and its connection to NetBeans; resulting in a large amount of time lost trying to resolve issues.

The register process was a valuable set of functionality to implement as it encompassed a lot of what was learned in the university semester. Although it was fairly standalone in the project, where we could use premade xml's for the other functionality, so we were not reliant on registering a user on every instance.

The functionality in search was challenging to implement whereby initially I had followed the similar lab's logic however upon discussion with the group, the design was changed to make the flow better for the user in our opinion. Eventually I ended up only doing the basic part of search, while the more in depth functionality was taken over by Harry and developed further.

The most difficult but also most valuable aspect of the actual coding attempted, in my opinion, were the REST and SOAP services. They proved to be very challenging for myself, requiring help from other group members as we worked together through the development of the entire section. I feel it is the aspect of the project which has the most potential for future application and thus all knowledge gained about it is beneficial. However, in order to implement REST and SOAP, an understanding of the flights.java, flight.java, users.java and user.java was necessary, as well as the jersey jars' functionality. Although we did not complete REST and SOAP due to time constraints before the deadline, we will work to finish it after the deadline due to this seen value.

Harry

The data structure of the assignment was relatively simple at the start, however got more complex as we continued. As an example, there were issues in the way things were written to the xml file, requiring a new class to hold the list of seats, however these were quickly overcome. Having a constantly updated data structure was very beneficial in our ability to continue progressing through the project. There were some refactoring issues as a result of this, however were quick to be solved. There were some issues with the XSL Transform as the newer library had new practices I was not familiar with. To solve this I obtained an older version that had more documentation available.

I did a large portion of the commenting through the project, where when it was not immediately obvious what the method does, I commented a brief description of functionality. It is a good practice to comment code in the fashion as it helps group collaboration, to immediately be able to understand the work I have done, as well as allows me to not forget things I have done in previous development sessions. These are some of the few benefits of commenting throughout projects.

Handling the xsl aspect of the search function and writing to the xml files was another of my responsibilities. I felt that xsl is an effective technology to represent an xml document in an aesthetic manner. The fact that it is declarative instead of imperative meant it could be used without running the entire project for every test scenario. It allowed for faster prototyping and bug solving. There were some challenges with the input parameters, but once they were overcome this proved to be a great contribution to our project.

By covering the more complex aspect of java coding, the lines at which we developed each part of the project were a little blurred. Much of the business logic was reliant upon a lot of this code, thus it proved to be very important to our project. It was definitely the most valuable aspect as I enjoyed the fact that there is a way to control how a HTML page is displayed through the use of mixing languages.

As James mentioned, we worked together on SOAP and REST aspects, the sum of our experiences are highlighted by him in his reflection.

Andrew

Through the help of my team members we managed to complete one milestone at a time, and worked day to day to ensure we were making progress. This assignment has made it clear that teamwork and communication are the most essential factors in completing the required tasks. I personally struggled with understanding the logic at first, but can now see how the web services tie into the web design. I also struggled throughout the assignment with the syntax, as I have little to no background in programming, which has also improved through the course of the assignment. For example, for the requirement where non-registered users (viewers) search results, the results should show flights only and remain unselectable, whereas registered users would be able to book the flight by clicking on them. I found it difficult to understand in the first place, as the code to present the book now link (viewFlight.jsp) was in the flightSearch.xsl, which required a different type of if statement. Through a bit of online research and help from my team I was able to create the if statement and enter the correct parameters to the results and flightSearch page (isLoggedIn). Although it was tricky, after several attempts (errors and compile issues) the code was executed as expected.

Another issue I encountered due to my uncertainty in syntax was fetching a specific flight by ID. This code was necessary in order to display the selected flight to the customer in the viewFlight.jsp page. Again, with the help of my team they were able to assist in the syntax and correct execution of code. This was done by using flights.findByID. Another aspect of the project I did not fully understand was the xml, and how user login sessions were added. With the help of the in class labs I was able to create the loginAction.jsp page and use the userApplication.java object to unmarshall the users.xml. Although it is simple, it was a rewarding experience to see the code execute successfully when running the project. Overall, I think this project was very difficult and challenging, and really tested my ability to understand the code, design the code and bring it up to a functional level. I believe my contribution to the project would of been a lot better with the use of better time management techniques, as well as taking another subject prior to this to boost my knowledge in java.

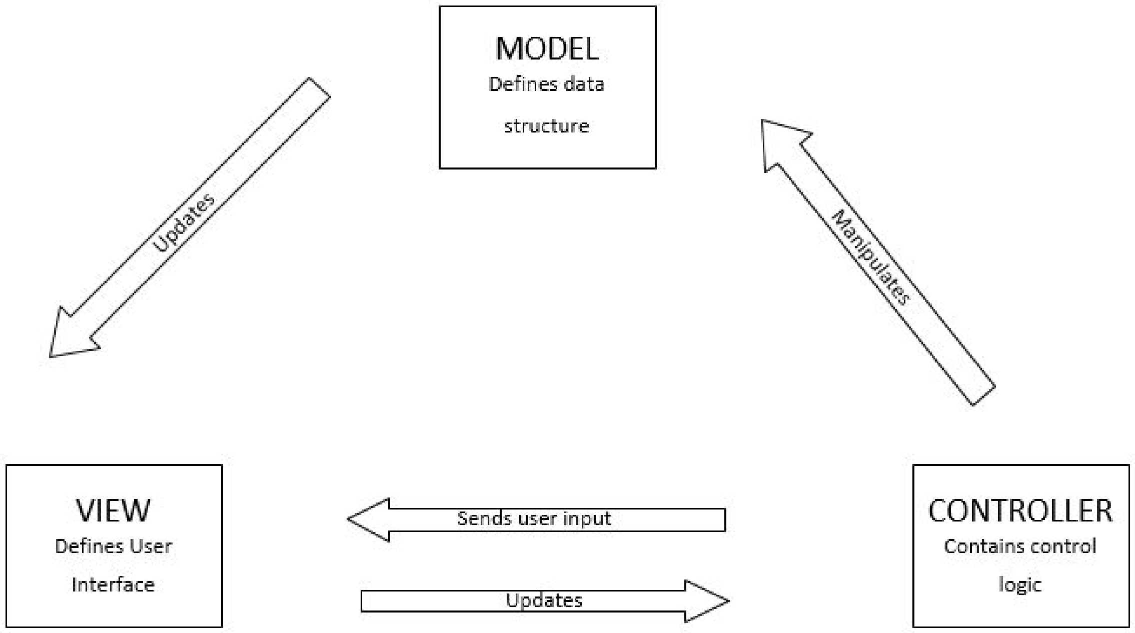
Issues and challenges in development of web applications and web services

Some of the main issues found in the development of web applications and web services are found within all forms of hardware and software development in computing. These include correctly identifying a project scope, producing the most up-to-date version of a program as well as allowing for update-ability and scalability in a project. Further, there are some specific issues in relation to web services and web applications that require the developer to use certain tools, design techniques and reference to online documentation as well as in-application support provided by coding tools.

Properly identifying the tasks within a project scope is essential in the process of developing anything in a web service or application environment. The mains reasons for this are to allocate resources and scheduling in the most accurate manner possible. If the scope is correctly constructed and understood by all team members, the project will have a much smoother construction process. This translates into less issues with project development flow, less conflict in working areas between members and less confusion as well as a clear path of progression through the project.

> *"Understanding the scope provides you with the foundations for managing project change and risk management. It enables goal setting and a timeline to work towards, as well as key points for reporting on the progress of the project"* (CIO Australia 2015)

To properly understand the scope, we made reference to MVC (Model View Controller) which is a software architecture pattern that partitions the application logic to optimise the flexibility of our design. Although our design does not fully represent MVC, it has been partially adopted in our project to maximise efficiency.



(Mozilla Developer Network 2016)

Model: This defines the data which our build contains, which is mainly flight details including flight seats, dates, cities and prices.

View: This refers to how the data is presented to the user (user interface).

Controller: Contains logic which updates the model by responding to user input whether it be a viewer, customer or admin. The model is updated uniquely depending on the user type. In our design the application classes are the controllers which handle this data.

Further, we utilised a trello board, as a tool to break down the assignment specification, creating a clear scope to better our understanding of the functionality from both a white-box and black-box perspective.

In building a project, as well as the final goal, changes in the computing industry must be considered. Using soon-to-be or currently obsolete coding languages/practices can create issues in the project's quality in the near future, as well as create potential conflicts of functionality between project aspects. The main reason to use old coding practices are that they usually have a larger online presences when researching documentation, as they have existed for longer periods of time. This can lead to people who are learning programming techniques to learn outdated methods, hindering their ability to produce relevant and up-to-date work. In developing our project; when researching aspects outside of the lab and lecture material, which was very rare, we were sure to only use sources that had been updated or created in recent dates.

Collaborative work is common practice amongst developers, and as such, we used collaboration in our problem solving within our group. Whenever issues were encountered we utilised group discussion, with all our members, to decipher the issue and develop a solution in order to ensure that no group member got stuck on an issue for too long a period of time. This collaborative solution allowed was effective as the knowledge areas of the group members showed to differ across the project, and the sharing of said knowledge improved all members' understanding as we progressed. Doing this ensured that all members had current knowledge of the processes and functionality present in our project. This became an advantage as we progressed when classes and methods became interconnected.

<u>GitHub:</u>
The issue of the need for collaboration in web service/application development is one that greatly increased development speeds when

solved. To remove the requirement for developers to be in physical proximity of each other in order to share projects or test each other's work vastly increased the rate at which they could work when separated. The tool "GitHub" greatly simplifies this process by storing the project online. In order to solve the issue of multiple people working on the project at once, it allows for users to push (upload) and pull (download) versions of the project. This, combined with the branching system (having multiple versions of the project running in parallel), allows for developers to progress with each other's work in a very efficient fashion. We used GitHub to take advantage of this efficiency, however it was not without complication. The coordination between NetBeans and GitHub was a costly issue for our group, in terms of time, as we faced difficulty with properly managing our Git branches and updates. It took some time for us to initially solve this issue through researching Git documentation in order to figure out NetBeans specific .gitignores that we needed to stop conflicts on every push/pull by a user. Once that was solved, we faced further issues with any imported Jar libraries and WEB-INF folder not being present in fresh pulls from GitHub. Eventually we were able to get past all the issues we faced, however the amount of time lost to this was significant.

Some Tasks we Faced Issues With:

Search:

Displaying the search results in the way we desired proved to be more difficult than first expected, with our understanding of the project scope skewed from what it turned out to actually be. After some thought, we realised it functioned much the same as the Login function, which we had previously constructed. This simplified the task as we understood it and allowed us to recreate it with much greater ease.

Menu:

Creating a global menu that displayed differently depending on whether a user was logged in or not at first seemed like a complex task, creating issues in our work process. However, we learnt to use the JSP scriptlets to access the session data in a single menu.jsp, constructing the menu based on that session data, before including that jsp in all the other web pages.

Seats:
Integrating seats of different classes into the flights was a scope requirement we did not expect, finding out through asking questions rather than interpreting it from the scope. Our initial project design had each flight being a type (Economy, Business) whereby all seats on said plane were of that type. Upon finding out about this, we had to overhaul our project to accommodate for this, using lists for each seat class, allowing the seats to be allocated to customers when booked.

Displays:
We had difficulties in the displays for the project, in regards to when certain display tools were to be used. We faced issues with functionality to being possible in the desired manner through the use of XSLT, consuming a lot of our project time. Upon discussing with our tutor, we concluded that we should just use java/jsp to return the displays that required functionality in the way we desired, while using XSLT in other display fields.

Styling:
To style the project to be consistent and neat across all pages is a timely process, which takes away from other development time. There are many tools to use which simplify the organisation and styling of web based applications. One of the more known of these is Bootstrap, which provides a detailed “grid system, that appropriately scales… predefined classes for easy layout options”(Mark Otto & Jacob Thornton and Bootstrap contributors 2017). This allows for a web service or application to be easy styled in such a way that incorporates multi-platform use, as well as making the styling process more efficient overall.

Troubleshooting:

Throughout the project, all members ran into several build errors which slowed our progression. Having a limited knowledge of the topics mean that it took quite some time to resolve these issues. The inbuilt NetBeans suggestions, provided upon pressing (Alt+Enter) were limited in their effectiveness, providing solutions to only a few issues. All other issues were resolved by group members, either through trial and error or looking at online resources like StackOverflow. The main use for StackOverflow we had is where people had encountered the error previously to us; a solution provided to that user would usually point us in the right direction to figure out how to resolve our issue.

Statement of Contributions

Contributor	Statement of Contribution
James Prevedoros	**Wrote:** ● **Page descriptions** ● **Own reflection** ● **Issues and Challenges (based upon input from teammates)**
Harry Smith	**Wrote:** ● **Page Descriptions** ● **Architecture breakdown** ● **Own Reflection** ● **Provided input for issues and challenges**
Andrew Belekas	**Wrote:** ● **Introduction** ● **Page Descriptions** ● **Architecture breakdown** ● **Own Reflection** ● **Provided input for issues and challenges**

Bibliography

Mark Otto & Jacob Thornton and Bootstrap contributors 2017, *CSS*, viewed 29 May 2017, <http://getbootstrap.com/css/>

CIO Australia 2015, *How to define the Scope of Project*, viewed 27 May 2017, <https://www.cio.com.au/article/401353/how_define_scope_project/>

Mozilla Developer Network 2016, *MVC Architecture*, viewed 27 May 2017, <https://developer.mozilla.org/en-US/Apps/Fundamentals/Modern_web_app_architecture/MVC_architecture>