

FINAL REPORT

Shotmaniacs 2

Luca Fuertes Taweeapiradeemanee s-2932598

Thomas van der Boon s-2939347

Dinh Thuy Nhat Vy s-2803100

Duong Thu Huyen s-2985594

Daeun Kang s-2502569

Oliwier Szwalek s-2990091

Table of Contents

Project assignment 1:	2
List of MosCoW requirements + user stories	2
List of MosCoW requirements	2
User stories	5
Story Map	7
Prototype report	8
Project assignment 2:	9
Report on UML diagrams and SQL schema	9
UML use case diagram	9
UML class diagram	11
SQL Schema	13
Prototype	14
Project assignment 3:	15
Security analysis	15
Testing report	17
Project source code	18
Project changelog	18

Project assignment 1:

List of MosCoW requirements + user stories

List of MosCoW requirements

- **Must-have requirements:**

- client side:

- As a client, I want to be able to upload multiple events at once using an excel sheet [event name; event type; event date; event location; duration of project; client; client's email address & booking type] so that I can save my time and effort compared to manually creating each event.
- As a client, I want to have access to the available URL link containing all of the event details and fill in the form associated with the event so that I can provide necessary information and specific requests to the event organizer.

- Company Side:

- As a member of the company, I want to be able to login to the dashboard using the provided email address or a username with a password so that I can access the tools and the important information for my work.

- Administrator:

- As an administrator, I want to send the announcements [title; body; publisher; urgency; date & timestamp] to the whole crew or individual crew members so that they can constantly keep informed about the news of the company.
- As an administrator, I want to be able to assign roles and modify these permissions to the crew members so that they can access only the necessary features and data required for their job duties.
- As an administrator, I want to have multiple options to modify the bookings, such as adding/canceling a booking or removing the crew out of the event so that I can efficiently manage the events and ensure that the event is staffed appropriately.
- As an administrator, I want to be able to view the "crew" tab to see the bookings each crew has picked up as well as the number of hours that a specific crew member has spent on the event.
- As an administrator, I want to see a list of the latest incoming bookings so that I can stay up-to-date with the events' requests, which allows me to quickly review and prioritize new requests, ensuring that the company is providing timely responses to the customers.

- Crew member:

- As a crew member, I want to see the announcements/receive the email notification on time

so that I can always keep informed about the news as well as the changes made to the bookings that I've been assigned.

- As a crew member, I want to view the dashboard with my current enrolled bookings as well as a history of bookings that I used to work on, which will allow me to easily keep track of my upcoming assignments and past work, helping me plan my schedule accordingly.
- As a crew member, I want to be able to see a list of all events with a calendar view as well as the available bookings per date at any point in time so that I can assign myself to a booking that I want to work on, or know that the bookings has no crew spots left, which will be grayed out on the calendar.
- System designer:
 - As a system designer, I don't want to set a limitation on the number of different roles that each booking can have so that the event can be prepared to the highest standard possible.
 - As a system designer, I want to set a limitation on the number of production managers for each booking so that only one production manager can be assigned to each booking. Whenever there exists one production manager registering and assigning himself/herself to a booking, the option to register as a production manager will be grayed out and unavailable for all other production managers, which will help to ensure that there is a clear accountability for each booking and prevent confusion or miscommunication among the production team.
 - As a system designer, I want to make sure that there is a clear difference between an available option and unavailable one, by displaying the unavailable options as grayed out so that I can avoid confusion and ensure that the crew members can easily identify which options are currently available and which are not.
 - As a system designer, I want an email to be automatically sent to hi@shotmaniacs.com whenever a client has submitted a new booking so that the company will be notified as soon as possible when a new booking is received and can take appropriate actions to prepare for the event.
- **Should-have requirements:**
 - client side:
 - As a client, I hope to have my own profile within the website, where I can view a list of all bookings that I have uploaded as well the corresponding crew members assigned to my bookings so that I can easily keep track of my events and the crew members working on them.
 - As a client, I hope to be able to cancel the event in case of any changes or unforeseen circumstances, which allow me to manage my events more flexibly and easily make changes to my events as needed without having to contact the event organizer directly.
 - Company side:
 - Administrator:

- As an administrator, I want to see the statistics about all events, including those that have a crew assigned or no crew assigned, so that I can effectively manage the staffing resources and ensure that the company is using its sources efficiently.
- crew member:
 - As a crew member, I want to have my own statistical page for my work with the filter option on clients and months in my dashboard so that I can easily find the specific information I need and better manage my own schedule.
 - As a crew member, I hope to be able to upload photo files to the website, under the folder corresponding to the appropriate name and date of booking so that I can easily share event photos with the clients as well as other team members and keep them organized for future reference.
- **Could-have requirements:**
 - Company side:
 - Crew member:
 - As a crew member, I want to receive the calendar invite in my email address when a booking is picked so that I can easily add the booking to my personal calendar and ensure that I don't miss any important appointments/events.
 - As a crew member, I want to send a de-enroll request to the administrator with a particular explanation so that I can move out without any serious affect to this booking and other crews, who are really interested in the event, can assign.
- **Won't-have requirements:**
 - Company side:
 - Crew member:
 - As a crew member, I hope to be able to view the scoreboard on the number of kaakjes for the "Kaakjes - Challenge" and update the counter after the booking has been finalized.

User stories

Client Side:

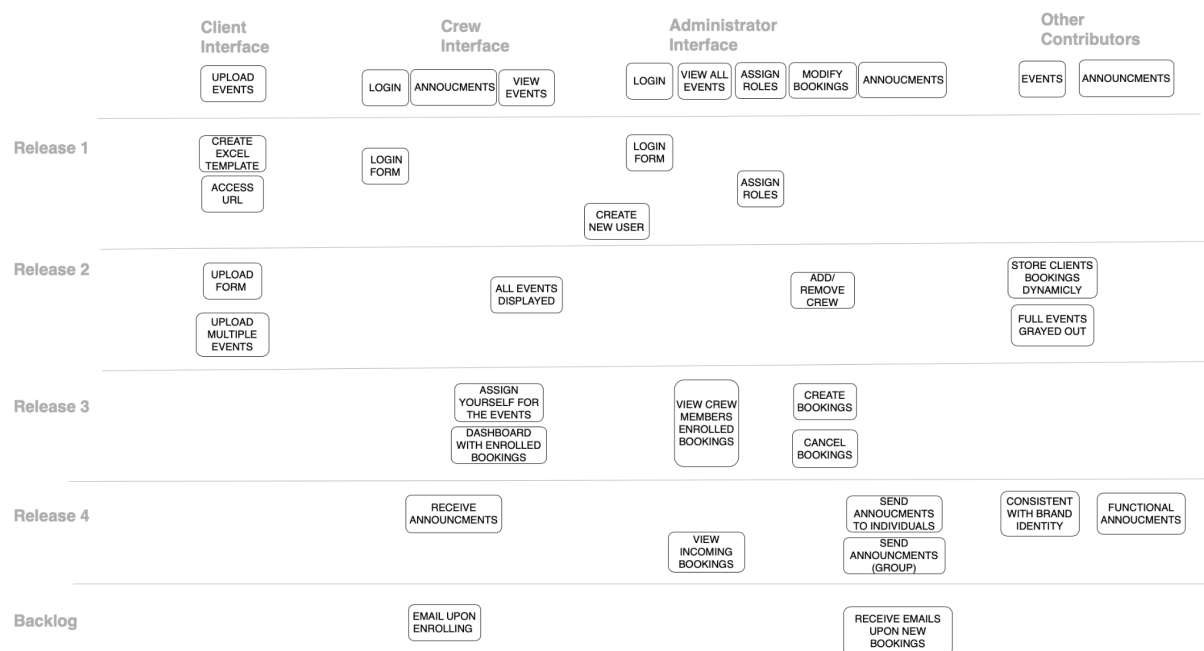
- As a client, I want to be able to upload multiple events at once using an excel sheet [event name; event type; event date; event location; duration of project; client; client's email address & booking type] so that I can save my time and effort compared to manually creating each event.
- As a client, I want to have access to the available URL link containing all of the event details and fill in the form associated with the event so that I can provide necessary information and specific requests to the event organizer.
- [Optional Extension]: As a client, I hope to have my own profile within the website, where I can view a list of all bookings that I have uploaded as well the corresponding crew members assigned to my bookings so that I can easily keep track of my events and the crew members working on them.
- [Optional Extension]: As a client, I hope to be able to cancel the event in case of any changes or unforeseen circumstances, which allow me to manage my events more flexibly and easily make changes to my events as needed without having to contact the event organizer directly.

Company Side:

- As a member of the company, I want to be able to login to the dashboard using the provided email address or a username with a password so that I can access the tools and the important information for my work.
 - **Administrator:**
 - As an administrator, I want to send the announcements [title; body; publisher; urgency; date & timestamp] to the whole crew or individual crew members so that they can constantly keep informed about the news of the company.
 - As an administrator, I want to be able to assign roles and modify these permissions to the crew members so that they can access only the necessary features and data required for their job duties.
 - As an administrator, I want to have multiple options to modify the bookings, such as adding/canceling a booking or removing the crew out of the event so that I can efficiently manage the events and ensure that the event is staffed appropriately.
 - As an administrator, I want to see the statistics about all events, including those that have a crew assigned or no crew assigned, so that I can effectively manage the staffing resources and ensure that the company is using its sources efficiently.
 - As an administrator, I want to be able to view the "crew" tab to see the bookings each crew has picked up as well as the number of hours that a specific crew member has spent on the event.
 - As an administrator, I want to see a list of the latest incoming bookings so that I can stay up-to-date with the events' requests, which allows me to quickly review and prioritize new requests, ensuring that the company is providing timely responses to the customers.
 - **Crew member:**
 - As a crew member, I want to see the announcements/receive the email notification on time so that I can always keep informed about the news as well as the changes made to the bookings that I've been assigned.
 - As a crew member, I want to view the dashboard with my current enrolled bookings as well as a history of bookings that I used to work on, which will allow me to easily keep track of my upcoming assignments and past work, helping me plan my schedule accordingly.

- As a crew member, I want to have my own statistical page for my work with the filter option on clients and months in my dashboard so that I can easily find the specific information I need and better manage my own schedule.
- As a crew member, I want to be able to see a list of all events with a calendar view as well as the available bookings per date at any point in time so that I can assign myself to a booking that I want to work on, or know that the bookings has no crew spots left, which will be grayed out on the calendar.
- As a crew member, I want to receive the calendar invite in my email address when a booking is picked so that I can easily add the booking to my personal calendar and ensure that I don't miss any important appointments/events.
- /*As a crew member, I want to send a de-enroll request to the administrator with a particular explanation so that I can move out without any serious affect to this booking and other crews, who are really interested in the event, can assign.*/*
- [Optional Extension]: As a crew member, I hope to be able to upload photo files to the website, under the folder corresponding to the appropriate name and date of booking so that I can easily share event photos with the clients as well as other team members and keep them organized for future reference.
- [Optional Extension]: As a crew member, I hope to be able to view the scoreboard on the number of kaakjes for the "Kaakjes - Challenge" and update the counter after the booking has been finalized.
- **System designer:**
 - As a system designer, I want to provide clients the option to upload a list of events in either CSV/xlsx or any other format in case they want to upload multiple events at once so that the clients don't have to fill in the booking form for each event, which can help them save their precious time and effort.
 - As a system designer, I don't want to set a limitation on the number of different roles that each booking can have so that the event can be prepared to the highest standard possible.
 - As a system designer, I want to set a limitation on the number of production managers for each booking so that only one production manager can be assigned to each booking. Whenever there exists one production manager registering and assigning himself/herself to a booking, the option to register as a production manager will be grayed out and unavailable for all other production managers, which will help to ensure that there is a clear accountability for each booking and prevent confusion or miscommunication among the production team.
 - As a system designer, I want to make sure that there is a clear difference between an available option and unavailable one, by displaying the unavailable options as greyed out so that I can avoid confusion and ensure that the crew members can easily identify which options are currently available and which are not.
 - As a system designer, I want an email to be automatically sent to hi@shotmaniacs.com whenever a client has submitted a new booking so that the company will be notified as soon as possible when a new booking is received and can take appropriate actions to prepare for the event.

Story Map



According to the story map, that is composed by user stories but in this map focusing on the functionality of web application. In the story map, it is divided into four main things which are: client, crew, administrator and other contributors. Each component has main functions to focus on the upcoming weeks.

From the client side, it required an application form that the client can apply the event to the company. Based on the user stories; create the excel template, access URL, able to upload form, able to upload multiple events. Those functionalities are required from release 1 to release 2 to finish.

For the crew interface, it required login, announcement, and view events in their interfaces. According to the user story, the crew should get their company account to login on the website. Then crew members should be able to view the upcoming events and history of events after they login. Also, crew members should be able to enroll in the bookings when they still have slots. The crew member was also able to see the announcement that the administrator published it. After mandatory functionalities are done, make functions for crew members could be able to receive the email for the upon enrolling.

The administrator interfaces required more than crew member side, because more functionalities would be added. For the login form might be the same as client side, the system might contain the authentication to recognize which account is administrator and crew members. The view all events function, it shows the crew members enrolled in the bookings with statistics. The administrator can be able to view the incoming bookings from the client side, therefore they could decide to approve or not.

Thus, the administrator can be able to modify bookings like: add / remove crew members in the bookings, create bookings, cancel bookings. The final requirement is to publish the announcements on the crew side. The administrators were able to send the announcement to individually and a group of crew members. In the backlog, the administrator should receive the email that is related to the upcoming new booking.

For the other contributors, there are two factors: events and announcements. For the events, it should be able to store the data dynamically when clients submit their application form. The crew and administrator could be able to see when the booking's slot is full. For the announcement, it should be able to be a functional announcement.

In conclusion, all the appearance in the website should represent the brand identity of the company.

Prototype report

The designs are read from top to bottom, every image showing either progression, as a step in the application, or a new page within the application. Some designs are unfinished or will be changed in the future/final design. Some other parts of the design are optional and will be implemented depending on the time available.

Crew member

The diagram illustrates a dashboard layout for a system. The top navigation bar contains the 'Company Logo', 'Dashboard', 'Calendar', and a user profile icon with the text 'Log in info'. The main content area is divided into three sections: 'Contender view' (a calendar grid with a 'Report clash' callout), 'Current Bookings' (a list of bookings), and 'History of bookings' (a list of past bookings). To the right of the 'Contender view' is a sidebar with 'Name of event', 'Descriptions', 'Roles already taken', and 'Available role'.

Client

Company Logo	Application Form
<p>One Event</p> <p>Name of event: _____</p> <p>Event type: _____</p> <p>Date: _____</p> <p>location: _____</p> <p>Duration: _____</p> <p>Client info _____</p> <p>_____</p> <p>Booking type o photography o film o Marketing o others</p>	<p>Multiple</p> <p>Provide excel form _____</p> <p>Submit here</p> <p>_____</p> <p>_____</p> <p>_____</p>
<p>Submit</p>	

Administrator

The wireframes are organized into several main sections:

- Crew / Announcement**: A header section with two tabs labeled "Crew" and "Announcement".
- Events**: A section with a title "Events → crew assigned" and a list item "Event Name, Role...". Below it is a large empty box.
- Latest incoming booking**: A section with a title "Latest incoming booking" and a list item "o". Below it is a large empty box.
- Calendar - view**: A section with a title "Calendar - view" and a table structure. The table has 4 rows and 6 columns. The first two columns are empty. The third column contains horizontal lines. The fourth column contains horizontal lines. The fifth column contains horizontal lines. The sixth column contains horizontal lines.
- Booking info**: A section with a title "Booking info" and a list item "o". Below it is a large empty box. A button labeled "REMOVE" is located at the bottom right.
- Crew Modification**: A section with a title "Crew Modification" and a list item "o". Below it is a large empty box.
- Current Announcements**: A section with a title "Current Announcements" and a large empty box.
- New Announcement**: A section with a title "New Announcement" and a form with fields for "Title:", "Body:", "Publisher:", "Urgency:", and "Date + time stamp:". Below the "Body:" field is a large empty box.

Report on UML diagrams and SQL schema

UML use case diagram

UML use case diagram



The use case diagram represents how particular stakeholders are interacting with the web application, exploiting its potential. We differentiate three types of stakeholders, who are utilizing the site's functionality, the Shotmaniacs admin, Shotmaniacs crew member and the client.

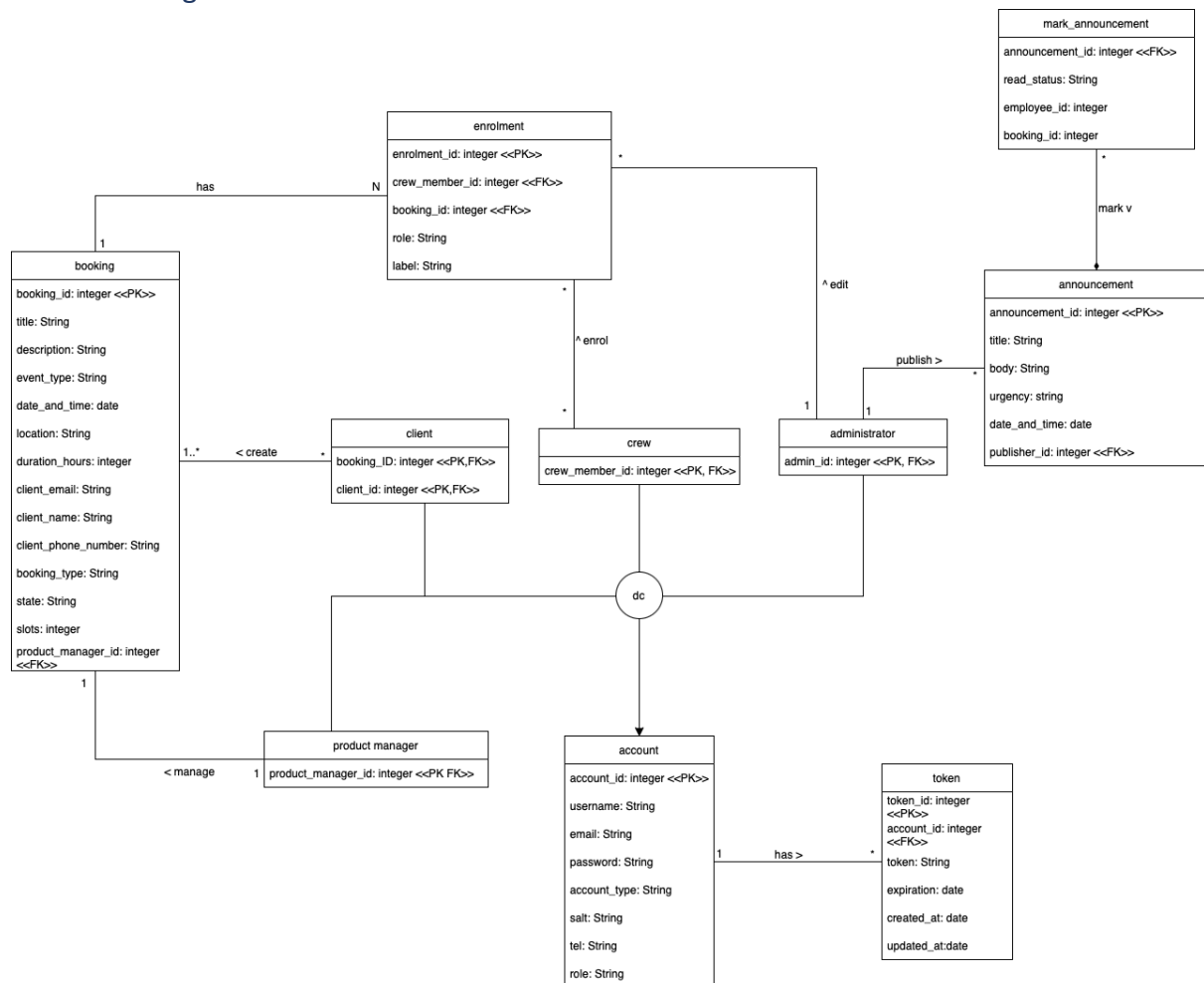
The admin is the person with the most diverse set of activities to exploit in contracts with other stakeholders. The amount of possible tasks is caused by the admin's responsibility to manage the system. They have the authority to manage crew members, specifically access personal information, add or remove affiliates from the event or manipulate their roles. The admin is also given the opportunity to accept or decline event bookings, create announcements that are shared with the crew.

The actor crew member represents Shotmaniacs associates. The developed web application grants them with instant visibility into submitted bookings on the site and that

results in possible enrollment for an event of interest. Beforehand crew is capable of accessing specific information and resources about the occasion, searching for a booking of choice, changing the label of event depending whether it is in progress, under review or done, and filtering received announcements for the one that were already read and new.

The actor client impersonates a potential end user of the web application. The user interested in submitting an event to gather a specific crew has the authority of inputting one singular event or multiple events at once with the use of provided excel template. However in order to gain such an opportunity a user has to create an account, which will store his personal information and ease the process of creating an event.

UML class diagram



The diagram illustrates the connection between each data class in our web application. It can be divided into five main categories to explain.

First we have an account class, this class stores the data related to the login and personal information. Also, this class connected with the token class, which stores the token and the data relevant for the token. One account can have many tokens when they log in to the website multiple times. For the account class has subclasses which are: product_manager, client, crew_member, administrator.

The product_manager class is only related to the booking. That is because the booking class contains the product_manger_id because of the requirements. According to the requirements each booking has one product manager. The result in the product_manager class contains the product_manager_id which is foreign key from an account called account_id.

The client class can make an application form that is related to the bookings, which means the data in the booking class is based on the client side. Therefore, the client class contains the booking_id and client_id. Both are the primary key for the client class and foreign key that came from the booking class and account class.

In the booking class, this class stored the data more than other classes. This is due to the fact that the client should fill the form of application about the event. So, the company can identify what type of event, location, time duration and others to make a booking on the crew side.

According to the diagram, the enrollment class has three relationships with booking, crew and administrator.

For the relationship between enrollment and booking, the events need participation of crew members. That means the enrollment is the list of the number of crew members that participate in the specific events.

Between enrollment and crew class, for this point crew members can select the booking and role in the event.

For the last relationship between administrator and enrolment class, administrator can edit the list of the crew member in the enrolment class. Which means the administrator can delete or add other crew members in the event.

For the label in the enrollment diagram, that is the status of the event on the crew and the administrator side.

For the administrator functions, they can publish the announcement on the crew side. Therefore we have an announcement class that connects to the administrator class.

In the announcement, it contains announcement_id for the primary key and publisher_id for the foreign key that is from the administrator class. Also, the class of mark_announcement is a component of the announcement class. The mark_announcement class contains the announcement_id for foreign keys. The remaining data is related to the status of the read by the crew member.

SQL Schema

enrolment(enrolment_id PK, crew_member_id, booking_id, role, label, FK crew_member_id REF account(account_id), FK booking_id REF booking)

mark_announcement(announcement_id PK, read_status, employee_id, booking_id, FK announcement_id REF announcement, FK employee_id REF account(account_id), FK booking_id REF booking)

announcement(announcement_id PK, title, body, urgency, date_and_time, publisher_id NOT NULL, FK(publisher_id) REF account(account_id))

token (token_id PK, account_id NOT NULL, token, expiration, created_at, updated_at, FK account_id REF account)


account(account_id PK, username, email, password, account_type, salt, tel, role)

booking(booking_id PK, title, description, event_type, date_and_time, location, duration_hours, client_email, client_name, client_phone_number, booking_type, state, slots, product_manager_id NOT NULL, UNIQUE(product_manager_id), FK product_manager_id REF account(account_id))

Based on the provided schema, it is evident that it was derived from a class diagram and transformed into an SQL schema. The majority of the classes from the class diagram are present in this schema, with the exception of the account class, which exhibits disjoint relationships. This disjoint relationship arises due to the existence of subclasses for the account class, namely: product_manager, administrator, crew, and client. Consequently, these subclasses have been consolidated into the account class, and specific identifiers associated with each subclass are referenced to the account class.

Prototype

Client side

 APPLICATION FORM [You need help?](#)

One Event

Event Name

Event Type ☐ Club Photography ☒ Festival ☐ Product Shoot

Event Date

Location

Booking Type ☐ Photography ☐ Film ☐ Marketing
☒ Other

Duration

Name


Email

Tel.

SUBMIT

Multiple Events


STEP 1: Please download the given template

 [Application Template](#)


STEP 2: Please fill in all event details given in the template


STEP 3: Please upload the filled-in template

No file chosen


 SHOTMANIACS

Company and client side for login page

 LOGIN [You need help?](#)

 SHOTMANIACS

LOGIN



☐ Stay signed in

[Forgot Password?](#)

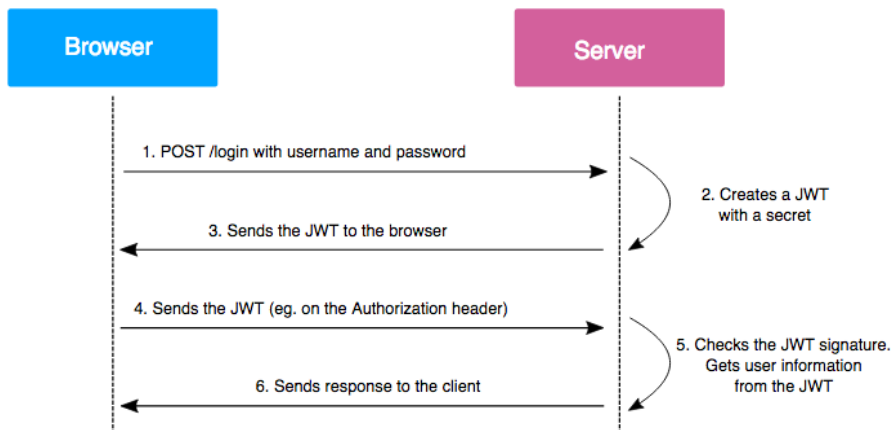
There are more than 40 pages for prototype, <https://www.figma.com/file/lsIWNN4v8rxNfYO0dlzjnH/Untitled?node-id=0%3A1&mode=dev> this is the link for the rest of the prototype for web application design. For this prototype we focused on the brand identity and easy to interact with the website when user use this website for the first time.

Project assignment 3:

Security analysis

Authentication of RESTFUL API Services

The method we have used to secure our API services is shown in the diagram below.



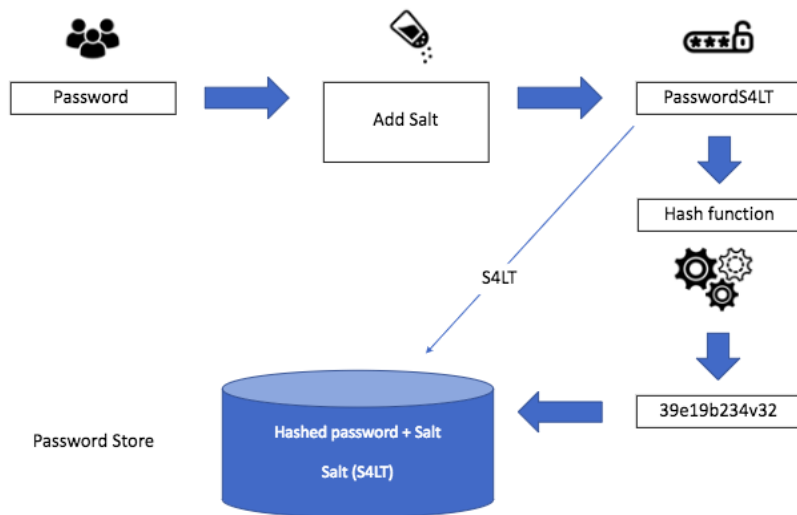
Each token contains an expiration date which by default is 1 day after token creation. This means every day, users will have to log back in which decreases the likelihood of unauthorized access.

The role is encoded into the token, allowing it to be checked in our RoleAuthorizationFilter class in order to deny or grant access to specific user roles. To clarify, the RoleAuthorizationFilter is our implementation of JAX-RS based authentication, allowing us to use a “RolesAllowed” annotation to specify access control for each API function.

The application stores the account ID of the client as well as the authentication token as a cookie. When we log out, these cookies are deleted for good measure. The token is included in every request to the API.

Passwords

We perform hashing and salting on all account passwords, and this is done automatically whenever an account is created. The hashing means we don't store plaintext passwords, and the salting helps protect the application against rainbow table attacks. The diagram below illustrates the hashing and salting process.



We considered the possibility of peppering our passwords for an added layer of security, but decided to focus our efforts elsewhere since other security aspects became more important once we had hashing and salting of passwords done. Additionally, as long as the hashing and salting is done securely, we believed that it would be sufficient for a project of this scope.

Injection Attacks and Other Vulnerabilities

We are quite confident that our server-side java code is secure, since we were consistent in using prepared statements to perform queries. However, we noted that most injection attacks would arise as a result of the client-side scripting with JavaScript. In this area we feel that there is some room for improvement in terms of protection. Most of the time we were consistent in implementing practices such as encoding data before it is submitted from the database. However, we identified a couple possible vulnerabilities which have since been fixed.

Testing report

	Client Interface	Crew Interface	Administrator Interface	Other Contributors
	UPLOAD EVENTS	LOGIN ANNOUCEMENTS VIEW EVENTS	LOGIN VIEW ALL EVENTS ASSIGN ROLES MODIFY BOOKINGS ANNOUCEMENTS	EVENTS ANNOUCEMENTS
Release 1	CREATE EXCEL TEMPLATE ACCESS URL	LOGIN FORM	LOGIN FORM CREATE NEW USER ASSIGN ROLES	
Release 2	UPLOAD FORM UPLOAD MULTIPLE EVENTS	ALL EVENTS DISPLAY	ADD/ REMOVE CREW	STORE CLIENTS BOOKINGS DYNAMICLY FULL EVENTS GRAYED OUT
Release 3		ASSIGN YOURSELF FOR THE EVENTS DASH-BOARD WITH ENROLLED BOOKINGS	VIEW CREW MEMBERS ENROLLED BOOKINGS CREATE BOOKINGS CANCEL BOOKINGS	
Release 4		RECEIVE ANNOUCEMENTS	VIEW INCOMING BOOKINGS SEND ANNOUCEMENTS TO INDIVIDUALS SEND ANNOUCEMENTS (GROUP)	CONSISTENT WITH BRAND IDENTITY FUNCTIONAL ANNOUCEMENTS
Backlog		EMAIL UPON ENROLLING	RECEIVE EMAILS UPON NEW BOOKINGS	

The table above shows what we have done after the 3rd sprint review. We are mostly focusing on the crew member side first after that go through the administrator side. That is because there are some overlapping functions between the crew and administrator.

According to the user stories, we are done with the interface for clients. Which client can upload a single event and multiple events based on the requirements. But sometimes, the URL is not working somehow, so we will focus on this problem to fix it as soon as possible. And also we need to consider UI design for clients that can easily recognize and access the website without explanation.

For the crew interface, we mostly finished the interfaces and functions but we still need to fix some problems like viewing the different sizes of the window. We figured out that problems are based on the margin and padding in CSS. That is because the descriptions are not the same length for every event. Except for those little problems the crew side mostly completed.

With administrator interface, except login interface, we are still working on the administrator interfaces and connect with back end and front end. For the administrator interface we will finish them in the upcoming week we are working on them. After that we will have some time to test them to make sure all requirements are functioning. It will take less than the crew side, that is because we already had experience with front/back end for implementing that we want to do for web application.

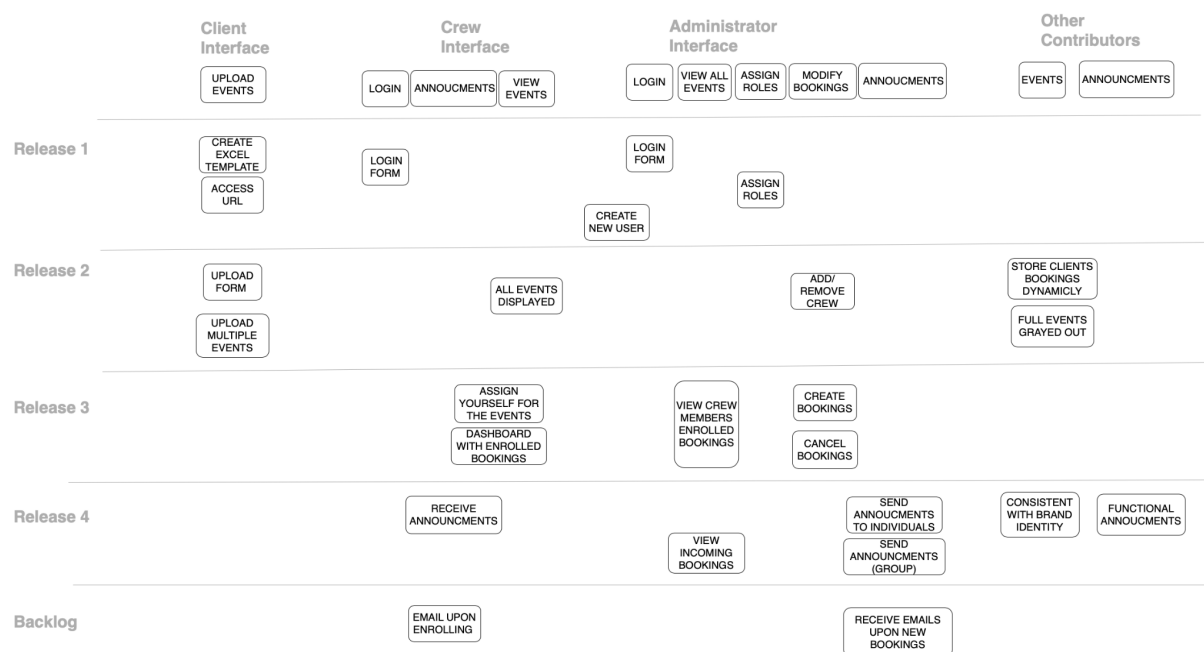
For the other contributors, those things are more related to the web design based on the requirements. We are done with those features when we implement the crew member side for web application.

Project source code

Project changelog

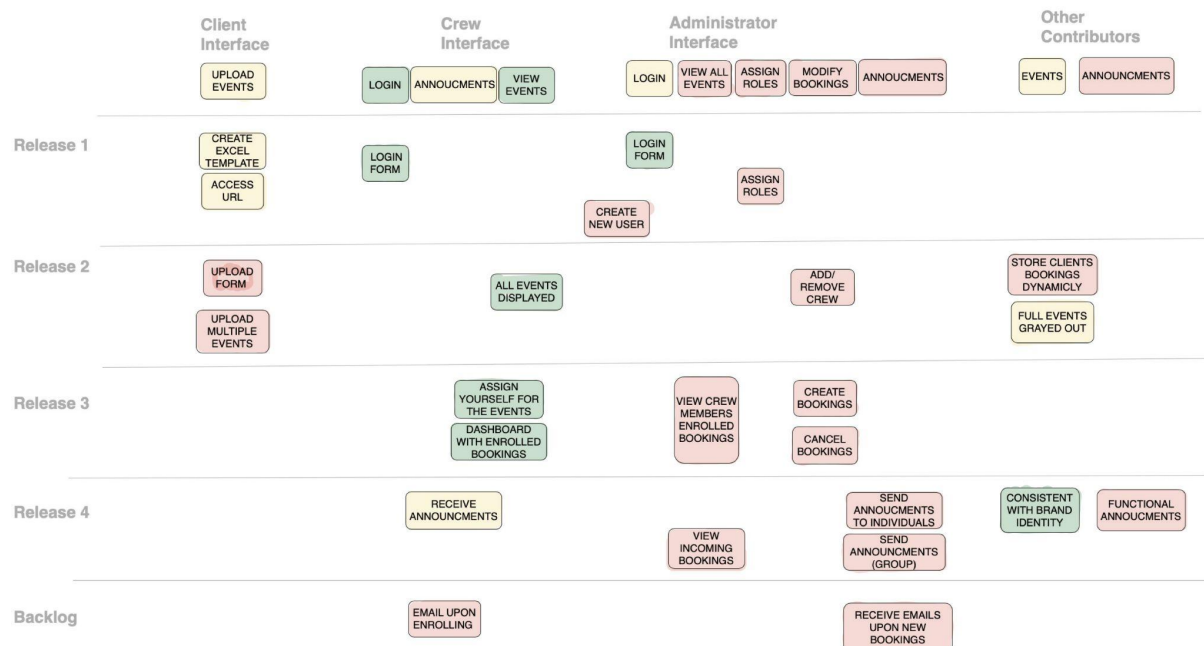
Release ONE

- Web design
 - design the low fidelity
 - choose the best design from the group member drawn
- Features
 - UML class diagram for database
 - UML use case diagram for interaction
 - generate story map from user stories



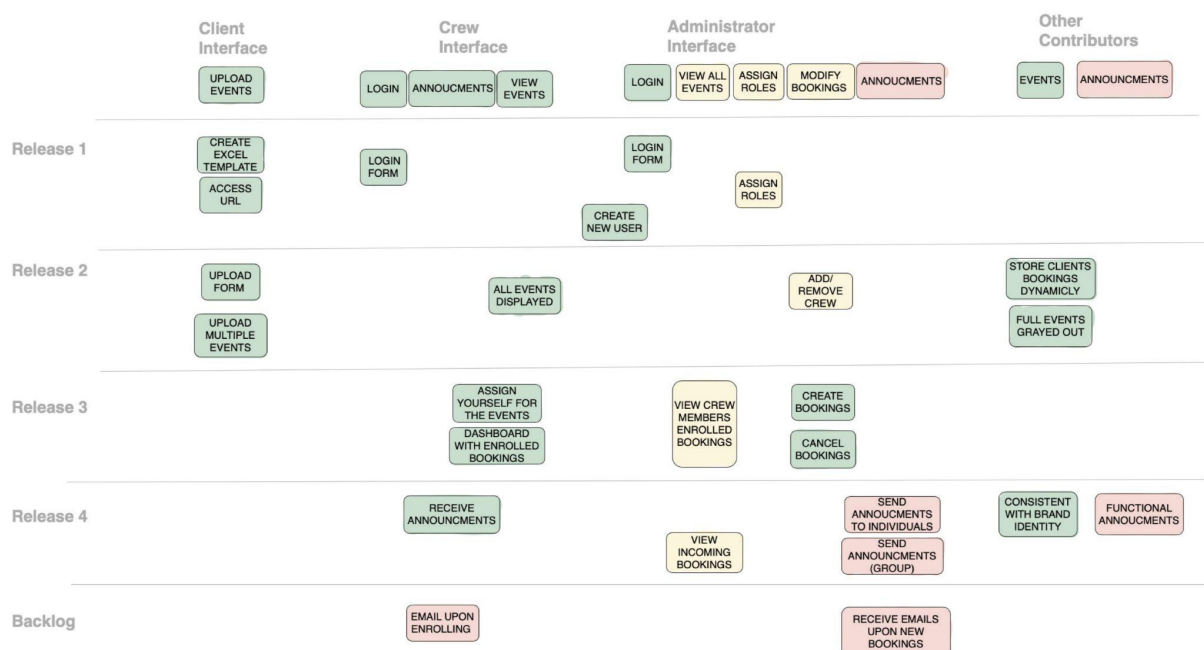
Release TWO

- Web design
 - based on the low fidelity design to make high fidelity design for the web
 - start to implement the front-end with HTML and CSS
- Features
 - set up the database and connect to backend
 - implement the RESTful API service
 - make the database be structured
- Security
 - save the secure data in the database for the password
 - implement the browser authentication



Release THREE

- web design
 - finished front-end for the client side
 - finished front-end for the crewmember side
 - start to implement front-end for the administration
- Features
 - mostly all the functionality in the back-end is done
 - make Junit test for check the functions work properly
 - most back-end functions that related to the client and crewmember side connect to the front-end
- Security
 - make authentication token for using in the login
 - sanitization of user input to make more secure



Release FOUR

- Web design
 - finished all the front end for administrator side
- Features
 - connect front-end and back-end for administrator
 - add more functions for the administrator
 - connect database when the web application deployed
- Security
 - make password salt and pepper to make more hard to attack
 - add Secure flag when cookie issued to the user
 - make token be disposable

