Joint Report Outlining Agile Development

Arthur Milner

Joseph Cauvy-Foster

Tommy Diclaudio

Strategy Planning

Here we carefully analysed the given specification, mainly viewing the project from the perspective of end users and the stakeholders, this is to ensure we develop a plan which satisfies both the stakeholder's standards and is accessible enough for end users to easily interact and utilise. The key focus point during this stage was to outline the software's required functionalities. For example, the permissions for different users, how booking costs are calculated, etc. Once we had a plan, we felt met the requirements we began to discuss the potential roles for each group member.

All group members had relevant experience in web development from the prior year, using this as a basis we gained a good grasp on each other's strengths and weaknesses. For example, Arthur felt he performed better with the back-end of his website project, so it felt appropriate to assign him with more tasks involving the database/logic of the software. Following the Agile methodology of remaining open to change in requirements, this stage of the process was more of a rough outline rather than a comprehensive guide. We also followed the Agile recommendation of stakeholder and developer working together to get iterative feedback during this stage to ensure we were on the right track using feedback from our tutor.

Continuous Team Iterations

Iterations on the project were generally conducted twice a week, however this was subject to change depending on the size of the tasks at hand and availability as we wished to follow the agile methodology of face-to-face meetings whenever possible. The frequency of the iterations is justified by the fact it allowed for enough work to review at each iteration and meant the project was constantly progressing.

The general outline of each iteration would be to review the outcome of the previous iteration and then create a plan to be implemented into the project by the next iteration. Of course, the beauty of this is that it allows thorough inspection at every stage of the development process, as should the review of a deliverable from a previous iteration be poor we can flexibly modify the plan for the next iteration, modify what was discussed previously, and then continue progress.

Examples of iterations would be an iteration for creating the log-in and home page GUI, then perhaps the next iteration might be connecting the two and adding a create booking GUI, whilst reviewing that the previously created pages meet a good standard. A further example of using iterations would be on the decision to implement MVC pattern, meaning a lot of code had to be modified and reviewed over multiple iterations.

Simplicity

The short nature of the iterations helped in managing the complexity of the system as the more manageable chunks allowed for the project to slowly build up, it also gave structure to the progress making it easy to know the functionalities currently expected to be implemented.

In attempting to further practice simplicity, the MVC pattern was adopted for the system, this makes it much easier to recognise what functionality happens where as its much more structured than the code without MVC. Since adapting the project to MVC it has become much easier to both code and follow. A further benefit is that it makes collaboration much easier as code feels more separated by its functionality/page.

Project Backlog

User	Functionality	Reason for Functionality	Priority	Status
Booking Staff	Login	To allow the software to identify who is operating it	Very High	Done
Booking Staff	To view screenings at a given cinema	To identify what screenings are on so they know what to book or give customers details of screenings at other cinemas	High	Done
Booking Staff	To create a booking for a screening	So staff can book tickets for cinema customers	Very High	Done
Booking Staff	To cancel bookings	So staff can cancel unwanted bookings for customers	Very High	Done
Admin	View Booking Staff details	To be able to identify the staff they might manage	High	Done
Admin	Edit Booking Staff details	So they can make necessary amendments/correct mistakes	High	Done
Admin	Delete Booking Staff	So they can remove staff who are no longer employed	High	Done
Admin	Generate Admin reports	So they can see how their different elements of the cinemas are performing	Medium	Done
Admin	View Film Details	So they can explain a film to a customer/check if it's on the database	High	Done
Admin	Edit Film Details	So they can make corrections to any film details	High	Done
Admin	Delete a Film	So they can remove films that are no longer showing	High	Done
Admin	Add a Film	So they can add new releases	High	Done
Admin	Edit Cinema Screening	So they can correct screening information such as the film or the time if there are delays	High	Done
Admin	Add Cinema Screening	So they can add more screenings for a film or perhaps even a new film	High	Done
Admin	Remove Cinema Screening	So they can update the screenings if a screening is no longer going ahead	High	Done
Manager	View Admin details	To be able to identify the staff they might manage	High	Done
Manager	Edit Admin details	So they can make necessary amendments/correct mistakes	High	Done
Manager	Delete Admin	So they can remove staff	High	Done

		who are no longer employed		
Manager	View Cinema details	So they can identify what values are stored about a cinema they might need to edit	High	Done
Manager	Add new Cinema	So they can add any Cinemas should Horizon expand their business	High	Done
Manager	Add new City	So they can add the city that any new cinemas might be in, including the new cities Cinema rates	High	Done

Note: Booking Staff as User also involves Admin/Manager and Admin as User also involves Manager