FINAL ASSESSMENT DELIVERABLE

31281 Systems Development Project

GROUP 19

- Daniel Ebrahimian 12961660
- Alex Munoz 12896940
- Bao Sheng He 12624401
- Jelisha De La Cruz 12003998
- Kazi Arman Asif 12859258

CONTENTS

1. Introduction

UTS uses systems such as Evenbirte and Ungerboeck to manage its seminars, as they are not suitable for small conferences since a lot of information needs to be provided. Therefore, a small system needs to be developed which will make it easier to arrange and hold simple seminars for the staff at UTS. We have been tasked with designing a simple seminar management system. We aim to plan, develop and document a simple system that requires minimal effort by occasional users to plan and run a seminar on campus.

2. Purpose of the report

This documentation will help to keep track of all aspects of our software and can improve its quality. It mainly focuses on the development and maintenance of the software. The purpose of this report is to outline, describe and present the key requirements of our software, how it works, how to develop it, how to maintain it etc. The following will be the main points of discussion covered in this report:

- Requirements Deliverables
- System Architectural Decisions
- UI/UX Decisions
- Database Construction
- Software Development Decisions
- Testing and Quality Control
- Team Management Process

3. Management Processes

Planning

Purpose of Planning

The reason that we plan projects is to ensure that the project is developed according to what was specified as well as taking into account potential risks and changes to specifications during this process. It also allows us as a group to display what our resources are, what knowledge we need and to clarify any assumptions as a group.

Context

As a group we are developing a seminar management system that caters to the client's needs. As stated by the client, the system needs to be similar to the systems that they are already using which are Eventbrite and Ungerboeck.

However, the proposed system must be specifically targeting the needs of UTS staff.

In developing this software we face a number of constraints, in particular a strict schedule, resources and knowledge or skills. As a group we have had to learn to work within these constraints so that we are able to deliver working software to the client on time.
As the time allotted for us to deliver the software has been about 12 weeks, we have to ensure that setbacks and delays are quickly dealt with in order to successfully deliver the software within that time period. As all of our team members have taken other subjects during this time period, it is up to them to ensure that they are able to manage their time between working on this subject and their other obligations.

As we are only students, we are not looking to use any resources that require to pay money or any sensitive details such as credit card details. Therefore our resources are limited to free or free trial based tools. We are also making use of resources provided to us as students of UTS.
Our team members come from different backgrounds and disciplines. We are constrained by our knowledge as we are not able to work on software that is beyond our knowledge and skill set. While we are able to learn throughout the course of this project, we are going to achieve the most success by using our knowledge and skills to develop this software.

This applies to the use of tools as well. While there might be tools that are better suited to the task, if the time taken to learn how to use the tool takes an unreasonable amount of time it would be better to stick with what is within our knowledge base.

Risks

When evaluating the risks associated with developing the software we decided to look at it from 3 different perspectives which were technical, project and knowledge. The reason that we assess the risks this early in the development process is to reveal potential issues before they occur. We also clarify the requirements of the project, as well as what is required of our team members in order to develop the software.

Technical

Some technical risks that are associated with the project include the use of new and unproven technologies.

There is also a risk with using outdated hardware, as it could potentially break down unexpectedly during the development process. The team could lose progress, data and other important information.

Project

The risks associated with the project include the scope, the schedule and the resource changes. The process of developing software can vary greatly throughout the process. There is no guarantee that the requirements of the client will be the same at the end as it was in the beginning. This poses another risk of the team not being able to keep up with the update requirements of the customer. This can make it difficult to figure out what features are considered in or out of scope which makes it hard for our team members to stay on track during development. This could lead to developing software that does not meet the client's requirements.

The schedule also poses another risk for the team. As the schedule is set in stone, if the team is behind in development means that rescheduling is required and can cause delays. There needs to be protocols in place in order to mitigate the damage and submit a satisfactory product by the appointed time. Time delays during the development process can be very costly and must be avoided.

Resource changes during the development process can cause major delays. Major changes to software and tools that are being utilised in the development process pose a significant risk to the development process.

Knowledge

Another aspect of the project that poses risks during development is our knowledge. While our team members bring in knowledge and experience, there is the potential for there to be gaps within our knowledge. For example, the use of new or new to the member tools can cause significant delays during the development process.

Another risk that knowledge poses is the clash of differing opinions from different members. Conflicts between members can arise if their knowledge on a particular aspect of the project clashes with each other and can cause uncertainty regarding what is the right course of action.

Planning Method

The way that we went about planning our project was to assess a variety of different facets of the problem that was presented to us by the client. As outlined above, we assessed the context of our members, our situation and how they would affect the way we would go about developing this project, and the risks that are associated with the development of the software. These evaluations allowed us to gather a better look on what resources we have,

and predicting any potential challenges we might face during the development. Then we can better mitigate these obstacles and reduce the amount of time and resources that would be used to resolve those issues.

We then looked at how we were going to tackle the workload. We decided to sort the proposed features (in the form of user stories) into a backlog, sorted by the provided priority level (Must have, Should have, Would be nice to have). Having the features sorted into priority on Trello lets team members easily see what needs to be done, and how important each task is. This forms the structure of our sprints, as our team knows what tasks should be done first, and what can be left until later.

Afterwards, we sat down as a team and estimated the efforts of each task to be completed throughout the course of development. While there are a variety of different estimation techniques, including t-shirt sizes, buckets and dot voting, we decided to go with scrum poker. Our team was the most familiar with scrum poker, and it tied in with our project management strategy of using story points. We found that the other techniques such as T-shirt sizes were not as detailed as scrum poker, as it utilises up to 13 values to measure the potential effort needed, whereas t-shirt sizes do not.

Purpose of Project Management

The reason that we manage the project is that many aspects of the situation will change as development goes on and we need to adjust the software and development processes as necessary. Some of these changes include the client's requirements, risks, constraints and assumptions. Moreover, project management also ensures that we are developing software that meets our objectives and the client's requirements, as well as keeping the team on schedule. If the team is falling behind the agreed schedule, it makes sure that the team knows what the protocol is for getting back on track for the delivery date.

Popular Methods

Waterfall Methodology or Linear Sequential Life Cycle Model

The waterfall methodology is an approach that is relatively more rigid than the agile methodology. Using this methodology the software development process is divided into sequential phases where each phase acts as the trigger for the following phase. These phases are the requirements, system design, implementation, integration and testing, deployment of system and maintenance. There is generally no overlapping between the different phases, as one phase cannot start until the previous phase has been completed and signed off.

Advantages:

- Allows for control and strict schedules for each phase of the project.
- Progresses through easy to understand phases
- Easy to manage as the phases are rigid
- Due to the rigidity it works well for small scale projects

Disadvantages:

- Difficult to estimate the time and cost for each phase
- It is very difficult to make changes to bugs found in the testing phase
- Not good for large or complex projects
- Not good for projects where the requirements are subject to change

Agile Methodology

The agile methodology is an approach that includes quick response to unanticipated issues and events that occur during the process of software development. The agile methodology consists of using incremental and iterative periods of time called sprints in which a specified amount of work is completed.

A sprint is a period of time allocated for a particular phase of a project. Sprints are considered to be complete when the time period expires. There may be disagreements

among the members of the team as to whether or not the development is satisfactory; however, there will be no more work on that particular phase of the project. The remaining phases of the project will continue to develop within their respective time frames.

Advantages of Agile model:

- Customer satisfaction by rapid, continuous delivery of useful software.
- People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other.
- Working software is delivered frequently (weeks rather than months).
- Face-to-face conversation is the best form of communication.
- Continuous attention to technical excellence and good design.
- Regular adaptation to changing circumstances.
- Even late changes in requirements are welcomed.

Few disadvantages of Agile:

- In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle.
- There is lack of emphasis on necessary designing and documentation.
- The project can easily get taken off track if the customer representative is not clear what final outcome that they want.
- Only senior programmers are capable of taking the kind of decisions required during the development process. Hence it has no place for newbie programmers, unless combined with experienced resources.

Project Management Method

The project management method that we have chosen to use for this project is the agile methodology. Due to its incremental nature, when features of the software need to be changed on short notice it can quickly be implemented. We all agreed that incrementally building on previous functionality with constant and consistent testing would be the best strategy for developing this software.

Team Formation

Members

1. Daniel Ebrahimian
2. Kazi Arman Asif
3. Jelisha De La Cruz
4. Bao Sheng He
5. Alejandro Munoz
6. Anzi Wu

Roles

Project Leader: Daniel Ebrahimian
Technical Leader(s): Daniel Ebrahimian
Librarian: Daniel Ebrahimian
Communications: Entire team
Quality Assurance: Entire Team
Documentation: Kazi Arman Asif

Team Management

Team Health

Individual

While we are working as a team we strive to have our members be satisfied with the project and its progress. We achieve through a number of ways.

If individual team members are unhappy with progress, or anything to do with the project, they can express this through our sprint retrospectives, or at any time during our meetings.

As our sprint retrospectives occur once per week during our standup meetings, if an individual feels the need to voice any concerns they can bring this up at any point on the Facebook Messenger chatroom, as this is where members are most accessible.

If these opportunities are not adequate for team members to voice their issues, after the tutorial there is an opportunity for members to have a private conversation with the project leader to voice their concerns and come to a solution.

As a team

In order to measure our team health we have decided as a team to have a health measurement form. This form allows us to rate how we feel we are going as a team and provides opportunity for individuals to add extra comments. We plan to complete this form on a weekly basis. These forms allow us to form a weekly snapshot of how the team is

feeling and can facilitate discussion at team meetings. We will utilize this insight to try and focus different approaches that are more harmonious to team members.

Communication

Team communication tools

Tool Name	Use	Justification
Facebook Messenger	This is where the group communicates about a wide range of topics not covered by other communication tools We also use this to host our weekly online standup meetings	All the members use this daily and have experience using it. Also has a group call function that other platforms are missing This tool allows members to communicate despite not being in the same area.
Slack	This is where the group conducts more formal messaging, as well as posting progress updates via the stand up channel	Is the industry standard for workplace collaboration Good for categorising different conversations into different sections
Trello	This is where our sprints are visualised. The group centralises the tasks that need to be completed during the sprint. These tasks have been organised according to their associated user stories.	Trello has a board and card system that makes it easy for members to see take in information without being overloaded by words Trello has been used members in other subjects and thus are able to use it without having to learn a new system

Google Drive	Hosts all files relating to the project, and where we all write the documentation. Also hosts a document which echoes the tasks displayed in Trello, but is also used to create our burn down charts, which shows our progress in comparison to how we should be progressing during a sprint. It does this through the use of Scrum points which are allocated to different tasks and reflect the completion of a sprint.	All members already use this platform to host files Includes a whole host of different tools that reflect what is usually done with applications such as those of the Microsoft Office suite, but includes the collaborative element of allowing multiple people to edit a document at once.
Github Issues	We use Github Issues to communicate technical issues to other members working on development	As we are already using Github to host our code, it only makes sense for us to use an associated program for communicating technical issues without clogging up the other channels.

Options Considered

Whatsapp

When deciding an informal communication channel, Whatsapp was one of our favorable platforms. The recent enhancements made towards group based communication with the enhanced performance of conference calls and video calls made compelling reasons to choose this platform. However, due to the ease of access our team members had towards Facebook Messenger, Whatsapp was not chosen. This is primarily due to team members not having Whatsapp accounts and apps setup whereas each team member already had Facebook Messenger.

Dropbox

When deciding a file sharing platform, we were hesitant to utilise Dropbox. Many team members raised concerns with this platform due to recent events relating to the security and integrity of the platform. The main concern resided within the broach of private data relating to user accounts and this information being widely available on the internet. This information included data such as usernames and passwords for logins to the platform.

Gitlab

Choosing our source control platform, it was a close decision between Gitlab and Github. We analysed our use case for source-control and concluded that we did not need advanced features such as Continuous-Integration or deployment services such as Kubernetes. We were after basic source-control features to primarily track commits and issues. For this reason we chose the more simplified platform, GitHub.

Expected Communication Behaviour

As a team, we utilise different communication platforms and methods to interact with group members. Each group member is expected and required to have access to:

1. Github
2. Facebook Messenger
3. Slack
4. Google Drive
5. Trello

When developing for this system, the developer is expected to make regular commits to Github. These commits are expected to be accpomanied with a short and concise message that describes the problem domain of the commit. Each modified file and addition should be related to this problem domain. The developer is expected to re-evaluate their commit if they are addressing more than one domain and break it down into smaller parts.

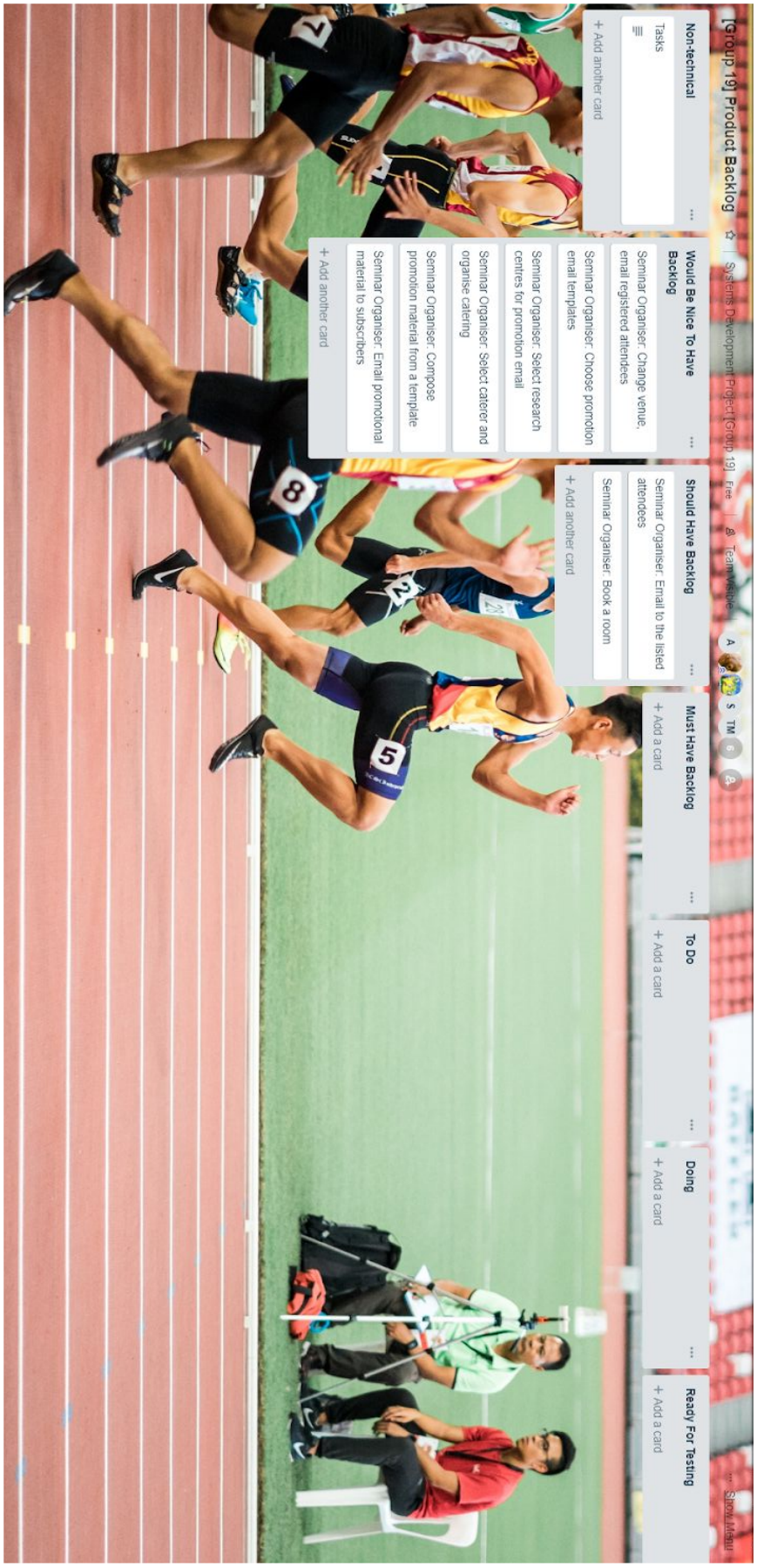
All informal text based communication is expected to occur within our Facebook Messenger group chat. Team members are expected to inform the group if they are unable to attend a meeting or to inform members that they require help with a section of their work. When conducting group calls/conferences, Facebook Messenger will be our first choice, however we will not limit our resources to just this platform. The first alternative will be a traditional phone call. In the event that none of these options become viable at a given point in time, a consensus must be attained to choose a new platform.

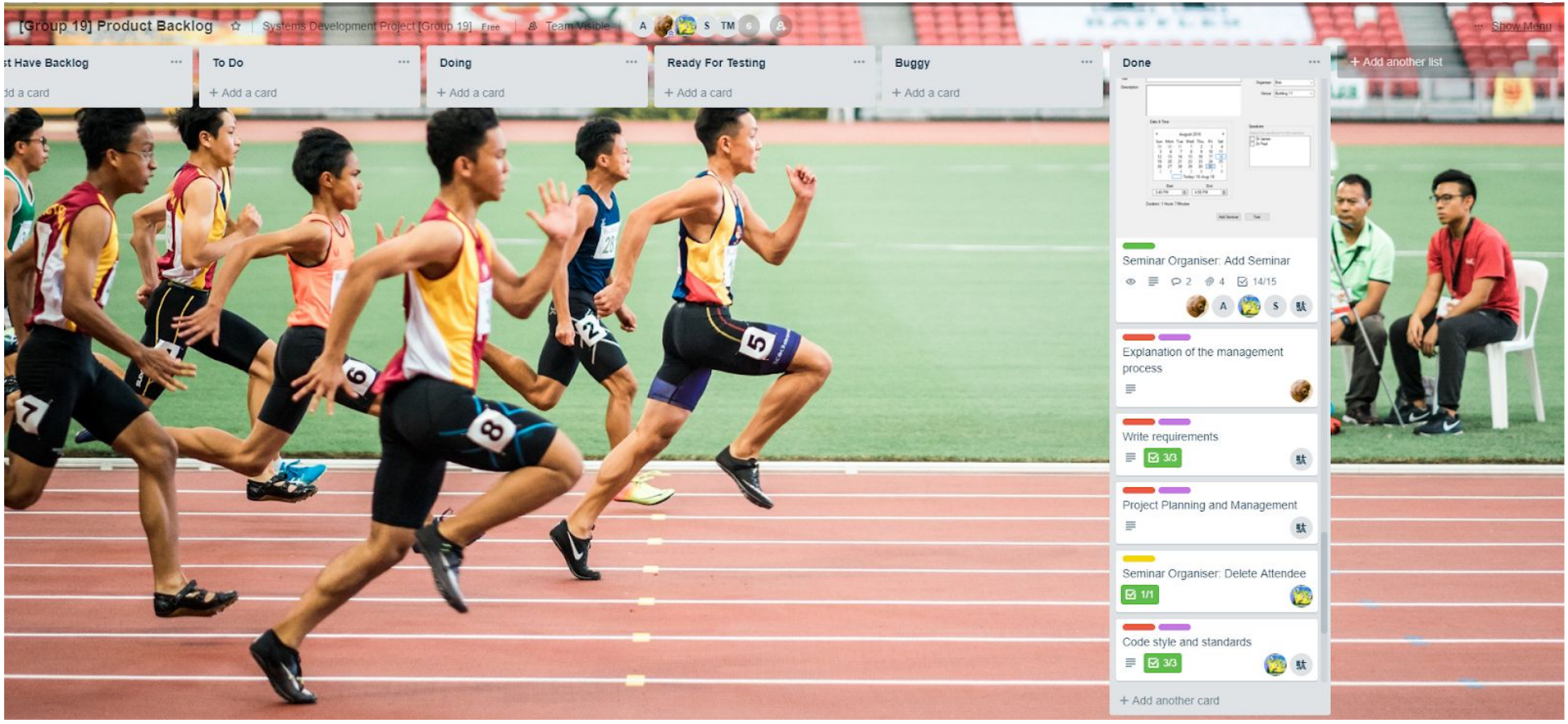
Slack will be used as our formal communication platform. Group members are expected to use Slack when they formally inform the group of their completion of a task. This is particularly related to our “standup” channel that is located in Slack. Peer-to-peer file sharing is expected to occur on Slack. This is to relieve the content clutter than can occur within Google Drive. Specific links that should be readily available will be distributed on Slack. Instructions on how to access critical infrastructure, such as Amazon Web Services will be communicated within Slack. Group members are expected to have Slack on their phone in order to receive important notifications and are expected NOT to silence notifications.

All file collaboration that is relevant to documentation and assignments is expected to be uploaded to Google Drive. Due to the nature of file sharing with teams, it is anticipated that the Google Drive will eventually become cluttered. We will have routine cleanups to ensure files are in the correct categories. Deciding when to cleanup the Google Drive is up to the discretion of the team to evaluate. Team members are expected to actively update their progression related to tasks in the ‘User Stories’ document. This document describes when tasks should be completed relative to sprints. Particularly, updating the status of a task between “To Do”, “Doing”, “Ready for testing” and “Done”. It is important to keep this data up-to-date as it is used to calculate the burndown chart. This information is further expected to be expressed in the Trello board.

Trello is expected to be used to visually express the team’s progression towards sprints. Our primary sprint board is named “[Group 19] Product Backlog”. All team members are expected to update their status of tasks here. Checklists are expected to be updated as relative tasks become completed within a Trello Card. Cards are expected to transition between “To Do”, “Doing”, “Ready For Testing” and “Done”. When features fail their testing stage they should be transitioned to the “Buggy” column and reasoning is expected to be provided in the comment section of a Trello Card. When clarification is required from the tester, communication should be initiated via direct messaging with the Slack platform. The creation of Cards and Assignees is expected to be maintained by the Scrum Master.

Screenshots of Trello Board




[Group 19] Product Backlog
Systems Development Project [Group 19]
Free
Team Visible
Show Menu
To Do
Add a card
Doing
Add a card
Ready For Testing
Add a card
Buggy
Add a card
Done
Add another list
Seminar Organiser: Add Seminar
14/15
Explanation of the management process
Write requirements
3/3
Project Planning and Management
Seminar Organiser: Delete Attendee
1/1
Code style and standards
3/3
Add another card

Project Schedule												
	Date:	9-Aug	16-Aug	23-Aug	30-Aug	6-Sep	13-Sep	20-Sep	27-Sep	4-Oct	11-Oct	18-Oct
		WK 3	WK 4	WK 5	WK 6	WK 7	WK 8	WK 9	WK 10	WK 11	WK 12	WK 13
No.	Release 0											
1	Sprint 0											
2	Sprint 1											
3	Sprint 2											
4	Sprint 3											
5	Sprint 4											
6	Sprint 5											
7	Sprint 6											
8	Sprint 7											
9	Sprint 8											

In order to be able to keep track of what needs to be done in order to develop a solution that suits the client's needs we used several different methods. These included bi-weekly meetings to supplement our progress and the use of programs such as Trello and Slack to showcase what has been done so far. During these meetings as a group we would decide what needed to be done based on their priority and how we would go about getting these tasks done.

SPRINT 0 12/8/18 - 22/8/18

During this sprint we decided to start working on the "Add Seminar" user story and broke it down into tasks that was delegated to group members. These were:

- Wireframes
- Database development and integration
- Front end development

The tasks we accomplished during this sprint were the implementation of the database and system, wireframes and front end development. We also started preparations on documentation and organising what needs to be submitted during the mid term. One of our members also took the time to read through the provided documentation and develop a deep understanding of what the client requires.

Challenges that were faced by the members during the sprint included having trouble focusing on tasks due to other subjects and jobs. Some members found it difficult to keep track of what task they were to complete during the sprint. There was also some issue with the front end development as the GUI was implemented before the wireframes were done. This resulted in the GUI having to be rearranged in order to match up with what was depicted in the wireframe.

Due to these challenges we resolved to spend more time focusing on our tasks for the sprint. We also found that in comparison to our progress in software implementation, our documentation has been lagging behind.

Sprint 1 23/8/18 - 29/8/18

Based on the results of our first sprint, we decided to adjust the way we ran our sprints this time around. This included creating a new excel document based on tasks that would be delegated and completed throughout the week. This document could be updated throughout the week by the members depending upon the status of their given task.

The tasks during this sprint were divided more evenly between the technical and non-technical categories. More members were put on to the task of writing out documents required for our mid-term deliverable which included an overview of our application, the purpose of our program and management processes. Meanwhile the other members were working on attendee related processes (register attendee, display attendee etc.) and integrating the database to the program via Microsoft SQL.

Our members faced several challenges whilst carrying out these tasks. This included implementational issues with our MVC architecture, time management issues and grammatical issues when writing the documentation. Overall the challenges regarding the clarity of tasks and time management have been largely reduced since the last sprint and hope to continue this trend in the sprints going forward.

Sprint 2 30/8/18 - 6/9/18

During our retrospective we found that the use of the excel document without tasks was beneficial to our progress throughout the sprint. However, there was an issue regarding the representation of assigned tasks between Trello and the excel Tasks document. Therefore for this sprint we resolved to have more collaboration between tasks displayed on Trello and the excel document.

As this is the last full sprint we have before the due date of the mid term deliverable, as well as the peer review presentation of our system architecture design, the majority of our focus was aimed on those two projects. Only two of our members worked on software related tasks with a reduced workload for the documentation. These included support for filtering the list of seminars by room, role support and user adding, editing and deleting functionality. The second of those two members was tasked with testing some features based on test cases outlined in the test strategies section of the deliverable.

As for the other members we focused solely on the documentation and the development of our deliverable. The tasks for this included:

- Reformatting the draft report
- Conducting research regarding different software development methodologies
- Conducting and documenting risk analysis for our software
- Documenting the scope of our software
- Designing test cases
- Analysing UI design and approving the most suitable choice for our software
- Drawing architecture diagrams
- Writing out the data table

Despite the improvement in productivity that we have seen over the past few sprints members of our team still faced challenges over the course of this sprint. One of these challenges was the due date of an assignment from another subject meant that some members were not able to focus on their allocated tasks during the sprint. Another challenge that we faced was the lack of clarity about the progress of a task. Our Trello board should reflect the progress of our tasks, however, it does not get updated as progress is made. Thus, the information shown on Trello is not accurate and does not reflect the true amount of progress that has been made. One of the major challenges we faced was the sheer amount of information regarding the software that we had to sift through in order to condense it into our documentation for our deliverable.

Sprint 3 19/9/18 - 26/9/18

During this sprint we redesigned the way that we used Trello, adding in more general tasks related to the project over tasks that were based on the user stories. The aim of this was to raise the team member's interaction rate with Trello.

During this sprint we set up the foundation of the final deliverable, as well as starting to fill out some sections including the introduction and project management. Other members were also working on the software, in particular implementing user roles into the system and changing the user interface based on the role of the user. There has also been work done on implementing the name tag functionality into the system.

Some of the challenges our team faced during this week was of time management. Some members found that the tasks they had taken on were a lot more time consuming than initially thought. Due to this, tasks that were taken on during this sprint were extended on to the next sprint.

Sprint 4 26/9/18 - 3/10/18

During this sprint we found that our way of using Trello was working well for the team. The team's interaction with Trello has greatly increased and finding out the progress of various tasks was much easier than it was previously.

At this point in time our software has had most of the main functions implemented. Thus, we started testing for bugs and enhancements to improve the system. At this time we also started acceptance testing in preparation for the final demonstration at the end of the semester. The only remaining missing functionality was the live database and printing of the name tags.

During this sprint along with testing, we had members continuing their work on the documentation accompanying the software. In accordance with our definition of done, team members who finished writing their sections contacted other members to sign off on the quality of the work done. During this sprint we also prepared for the presentation regarding the readiness of our system. As a group we met up and discussed what is to be presented and created the presentation slides.

Some challenges we faced was that as the end of semester was nearing, other assignments from other subjects deadlines were also quickly approaching, stressing out the members. This resulted in slowing the pace of our progress down greatly, as some of our members had multiple assignments and exams being conducted during this sprint. While unavoidable, it was agreed upon that we as a team should be managing our time a lot better.

Sprint 5 3/10/18 - 17/10/18

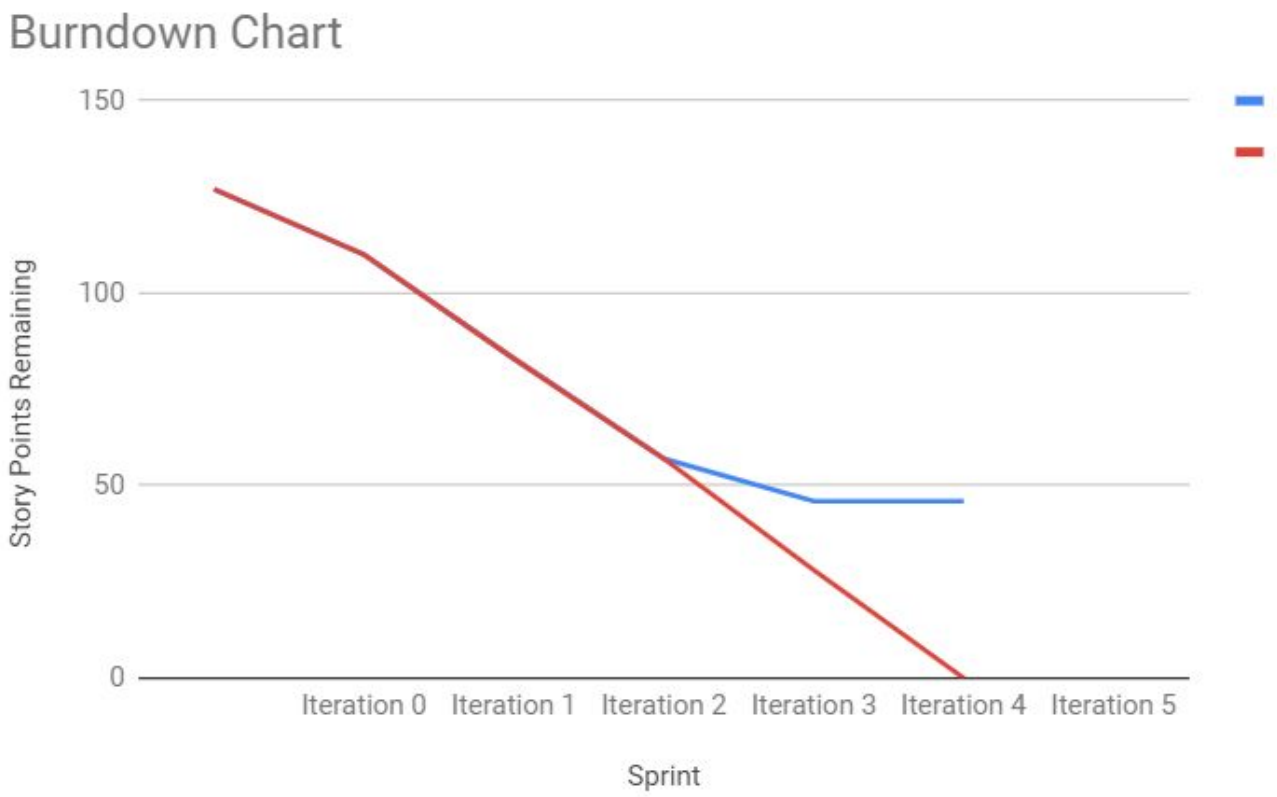
We have finally reached the last sprint of the project. While there were still a few issues with time management from members of the team, the overall workload has been reduced which has allowed us to work harder on this project.

Regarding the software, the issues with the database and the printing of the nametags has been resolved. At this point in time we have been rigorously testing the system for more bugs as well as conducting acceptance tests on our system.

Members have been working on the documentation, continuing their parts as well as starting on other parts that have not been completed and have not been assigned to a member. Other members have been reading through what has been written currently and editing.

A challenge that we faced throughout this sprint was that some members had final assignments due during this sprint, making them unable to respond to queries regarding the assignment until they were finished. This made communication difficult for some time, however, other members managed to step in and assist in their stead. Another challenge is the amount of content in the deliverable makes it difficult for members who are editing the writing to keep up with members who are adding new content. This makes the task of editing a daunting one, however it is something that must be completed. Another challenge that we faced was the lack of clarity regarding what is required in the deliverable. It was difficult to decide what elements are needed in the document against what is not needed.

Burndown Chart



The burndown chart is important as it provides the team with a great graphical representation of where we are at vs where we predicted we would be. This helps us to see if we are on track or if we are behind and should pick up the speed of development.

This burndown chart is a visual representation of our progress calculated using story points. The graph maps the Story Points Remaining against the Time we have left. The blue line is our current progress and the red line is the estimated progress we should have made. The reason that the blue line does not reach the bottom is because of the out of scope user requirements. We didn't complete all the optional requirements and hence our development ended with story points and tasks remaining.

Requirements

Purpose of managing requirements

The reason we manage the requirements is to ensure that the software that we develop does fulfil the needs of the client. Developing the software without managing the requirements increases the chances that the client will not be satisfied with the delivered product and breeds miscommunication between everybody. It also ensures that all team members and stakeholders are on the same page regarding the use and performance of the software. Additionally, the needs of the client are subject to change throughout the development process. This highlights the importance of managing the requirements and ensuring that all team members are working with the latest information at all times.

Managing requirements also allows us to expose dependencies between different requirements and gives us a timeline on when certain elements should be done and what order it should be done in. This structures the way that we approach the development and testing of this software.
We intend to achieve these objectives by using user narratives to manage the client's requirements, even in the cases where the client changes them.

In order to achieve the above objectives, we utilised several methods. These included implemented acceptance based development. Once we created a requirements backlog on our trello board, our developers use those cards as a basis of what functionality gets implemented and in what order. These cards are updated as we receive new information from the client regarding how they want the software to do.

Stakeholder Analysis

During the process of developing this software, we have had to keep in mind the influence that stakeholders have on the project. Thus we felt that it was important that we analyse the stakeholders and prioritise their wants and needs depending on the role they play in the project. Due to the small scale of the project, the members of the team make up the majority of the stakeholders. The stakeholders in this project are the:

1. Client
2. Developers
3. UTS Staff
4. Testers

<u>The Client</u>

The client (in this case the subject coordinator was representing this role) is the stakeholder who informs us of what the software needs to be and what objectives it

needs to achieve. The client is the first point of contact regarding how the software must run and is a representative of the future users of the system. He will also be the one to test the system when the software is presented. Thus, the client is a very important stakeholder to the project and it is important to prioritse their needs regarding the project.

Our communication with the client happens during the class time as well as online communication through Google drive. During class time we have the opportunity to request feedback for the current iteration of the program as well as answering any queries we have.

Developers

The developers are the members of the group who are involved in the actual development of the system. The developers are integral to the project as without them the software would not be created. They implement both the functional and non functional requirements that the client requests. Thus it is important to listen to their thoughts and opinions regarding the project as they know the software very well.

The communication with the developers happens at our weekly stand up meetings, the in-class meetings as well as through the social media we agreed to use for communication.

UTS Staff

The UTS staff are the predicted future users of the software, after it has been submitted to and tested by the client. We are designing the software for their use so it is important that while developing the software that we keep their opinions in mind.

Our communication has been through the written opinions of what they would want in the software that was provided to us by the client.

Testers

The testers are the members of our group who did not have a major role in the development of the software. We decided that once the developers felt that they had reached a milestone in their code, they would get one of the testers to begin testing that portion of the program in order to see if their are any bugs or defects. Thus they have played an important role in the development of the software and it is very important that we note down their observations regarding the software.

The communication with the testers occurs at our online weekly stand up meetings, the in class meetings as well as the times between those meetings through social media.

System Architecture

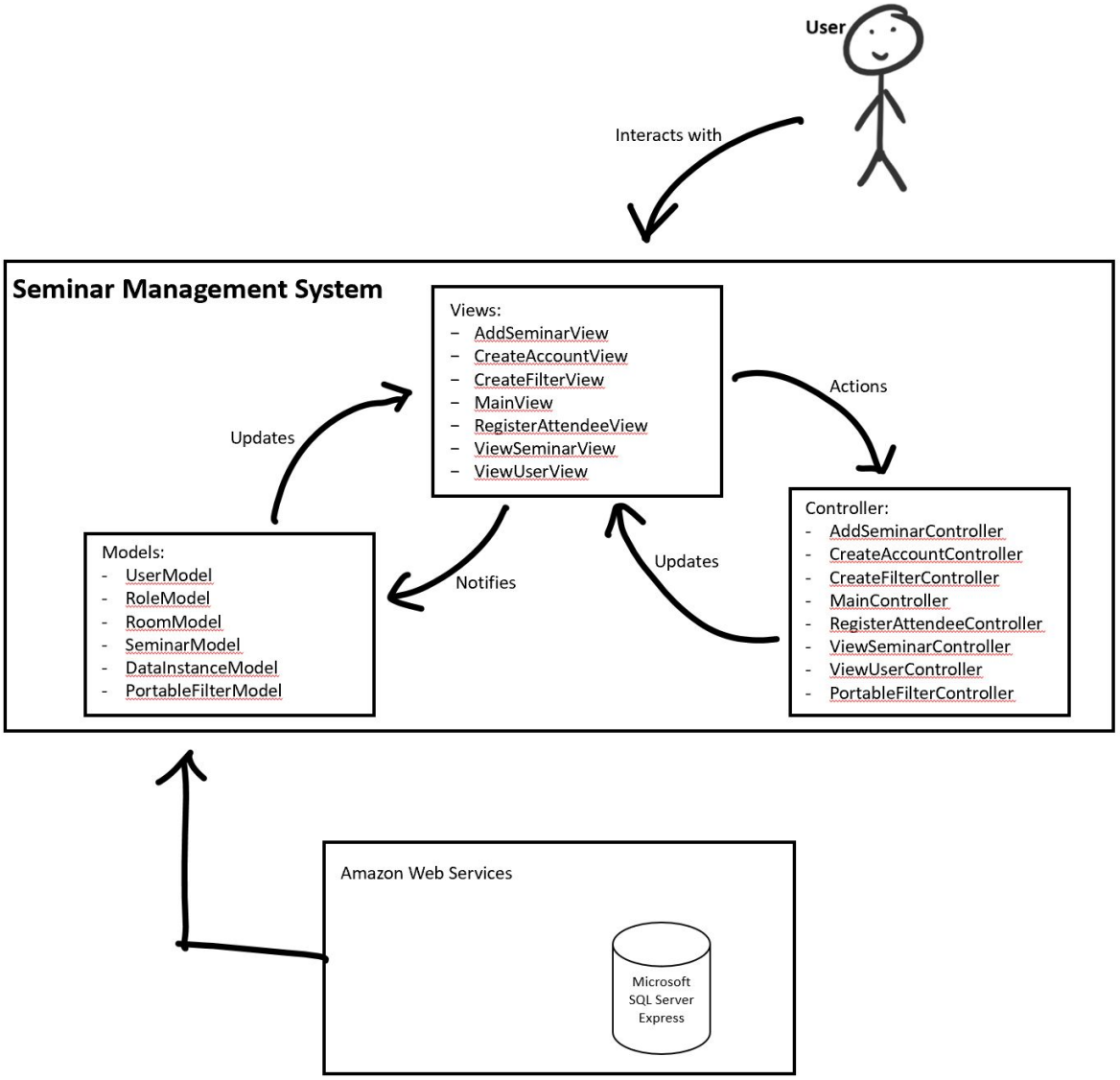
System architecture diagrams are a necessary way to give an overview of how the system will function to all stakeholders. It is important that it allows stakeholders to

see that their concerns have been met and that the system will provide the functions and behaviours necessary to address the business problem.

The Seminar Management System requires users to interact with an application that has the ability to display accurate information about current seminars. One challenge of designing such a system is how to deal with distributing data to multiple clients. For this requirement, we have chosen to use an external database that will be hosted in the cloud by Amazon Web Services. This database will be in Microsoft SQL Server format because our chosen development platform is Windows. By utilising a natively supported database format for Windows, we can be confident that we will be able to integrate the Database with ease.

The application will be made as a Windows Forms Application. Windows Forms Application is a common platform for Windows programming that easily integrates with our chosen database environment. The native application will be following the Model View Controller (MVC) design pattern to reflect the representations of the database better. This combination of system tools allows us to utilise a numerous amount of programming languages that are compatible with the .NET Runtime. However, we have chosen to use a single language explicitly; C#.

By developing a system architecture diagram we are able to clearly map out the components of our system and how they interact with each other. This helps developers to get straight into the development process and gives them a solid understanding of all the components.



This System architecture diagram provides us with a simple description of the way our software interacts with itself. For this project we decided that to keep the diagrams simple as the requirements of our project are not complicated and we wanted any stakeholders and developers to look at the above diagram and get a solid understanding of how our system works.

The file names indicate the intended purpose of that file and highlights the functionality and user requirement that it accomplishes. For example the AddSeminarView file displays a view that allows the user to add a seminar. This then sends any actions it has to the AddSeminarController etc.

ASSUMPTIONS

Reference	Assumption Item	Assumption Description	Reason
AS-1.0	Language	The users of the system will be familiar with the English language	The platform doesn't require support for multiple languages
As-2.0	Internet	The users of the system will have an active internet connection	The platform requires the user to be connected to the internet
As-3.0	Attendee are good people	It is assumed that the attendees won't perform malicious acts with the data exposed to them.	Customer provided feedback of the expected audience
As-4.0	Knowledge	Users have knowledge in using Windows computer	The platform doesn't have any help section.
As-5.0	Roles	One person can have multiple roles.	Doesn't restrict an attendee from also being a speaker or organiser.

Functional Requirements

Item #	Description
1	The system shall allow a user to login
2	The system shall allow seminars to be created
3	The system shall allow a seminar to be modified
4	The system shall allow a seminar to be deleted
5	The system shall allow new users to be created
6	The system shall allow users to change their details
7	The system shall allow attendees to delete their attendee record
8	The system shall allow user roles to be changed and allocated
9	The system shall allow seminars to be searched for by a filter
10	The system shall allow seminars to be viewed by all users
11	The system shall allow organisers to remove attendees
12	The system shall allow attendees to change their attendance status
13	The system shall allow organisers to remove attendees
14	The system shall allow organiser to see attendee details
15	The system shall allow organizers to book a room for a seminar
16	The system should allow attendees to attend multiple seminars
17	The system shall allow seminar organizer to email to the listed attendees

18	The system should allow the seminar organizer to email all the attendees for a nominated seminar
19	The system should allow catering to be organized for a seminar
20	The system shall communicate with a database
21	The system should allow promotional emails be sent to users
22	The system should allow organisers to create multiple seminars
23	The system shall let organizers request name tags be made

Non-Functional Requirements

Item #	Description
1	The platform should respond within a reasonable amount of time
2	The seminar information shall be readily available
3	The system should send print request for name tags to print servers
4	The password should be encrypted
5	The system should be scalable
6	The system shall be user friendly
7	The system should be easily maintained
8	The system
9	The system should have help feature
10	The system should have easy to learn interfaces

User Interface/User Experience Design

User interface design is the design of different kinds of applications with the focus on the user's experience and interaction, as well as the system usability. It can be prototypes with wireframes, or graphics designed by graphic interface tools. The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals, which is called user-centred design. A good user interface design helps user to understand the control easily. To be user friendly, it should obey basic rules such as clear alignment, understandable documentation, public consistency.

When creating our user interface designs we were mindful of the Jakob Nielsen's Design Heuristics (Jakob Nielsen 1995). As this system will be used by UTS staff who we assume do not want to spend a lot of time navigating or learning to use the system. Thus when we were designing the user interface of our system we referred to Nielsen's Design Heuristics to ensure that our system is user friendly and intuitive for new users. One heuristic that we kept in mind was making users recognize features on our user interface instead of recalling their use. This streamlines the user's experience of managing seminars within the system. We implemented this by keeping our button labels verbose as opposed to shortening them. Another example can be found in 8.1.2, where the fields are set out in a way most users would be able to recognize as a form.

We also aimed to have our software match between the system and the real world by having the words used when a user is creating a seminar and choosing a date, the screen provides a calendar for the user to select a date from. This view makes it easier for users to connect our system to the real world and thus find our system easy to use.

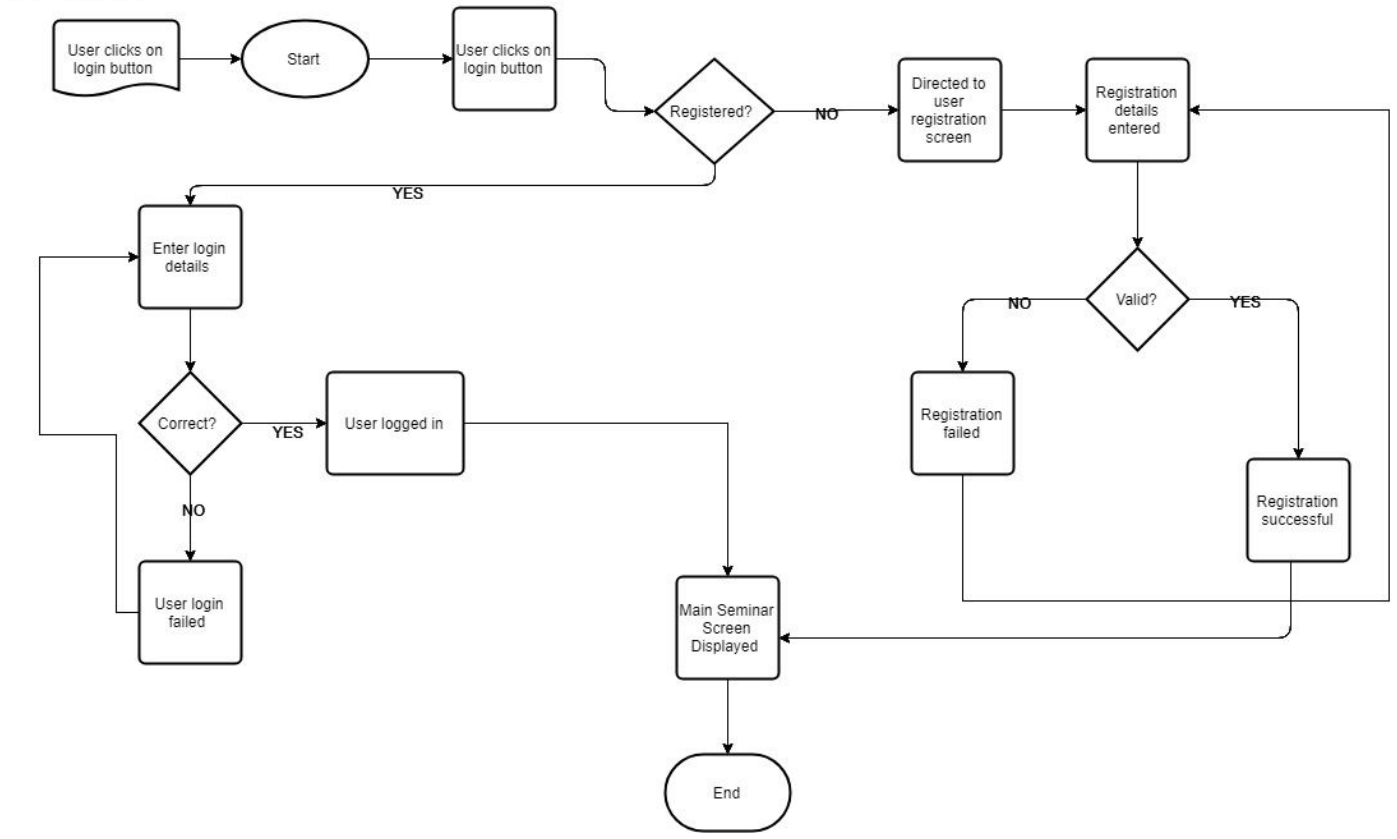
Flow Charts

A flowchart is a visual representation of the sequence of steps and activities needed to perform a process. Each step in the sequence is noted within a diagram shape. Steps are linked by connecting lines and directional arrows. With proper design and construction, it communicates the steps in a process very effectively and efficiently.

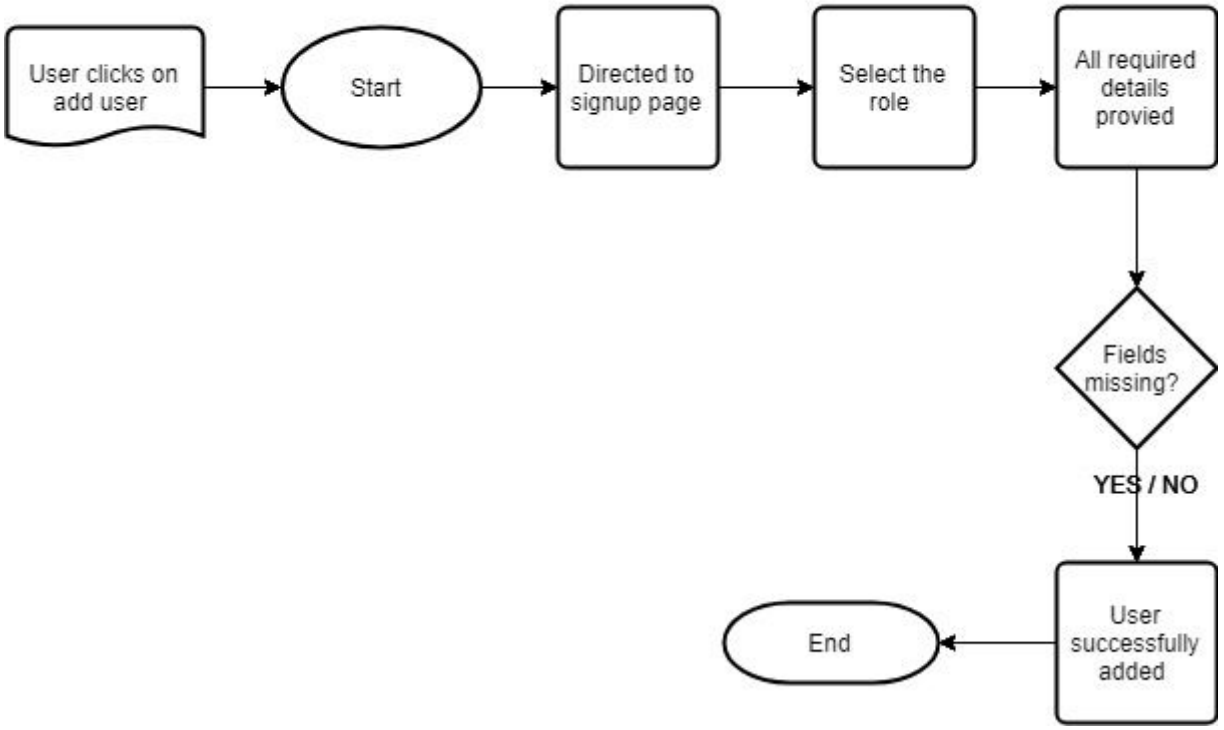
The reason we decided to include flowcharts in our report is because sometimes it is better to provide a visual representation to convey meaning as opposed to words. Flowcharts explain a process clearly through symbols and text. Moreover, flowcharts give us the gist of the process flow in a single glance. So, before we started coding or designing the system, we made flowcharts of the main activities within the system. This made our task much easier during designing the system since we already knew each of the steps in accomplishing a task. While creating the flowcharts we made sure we have our customer's needs set as the main priority and it was considered done only when all the members agreed that the processes in the flowcharts fulfilled all customer requirements.

Flowchart Diagrams

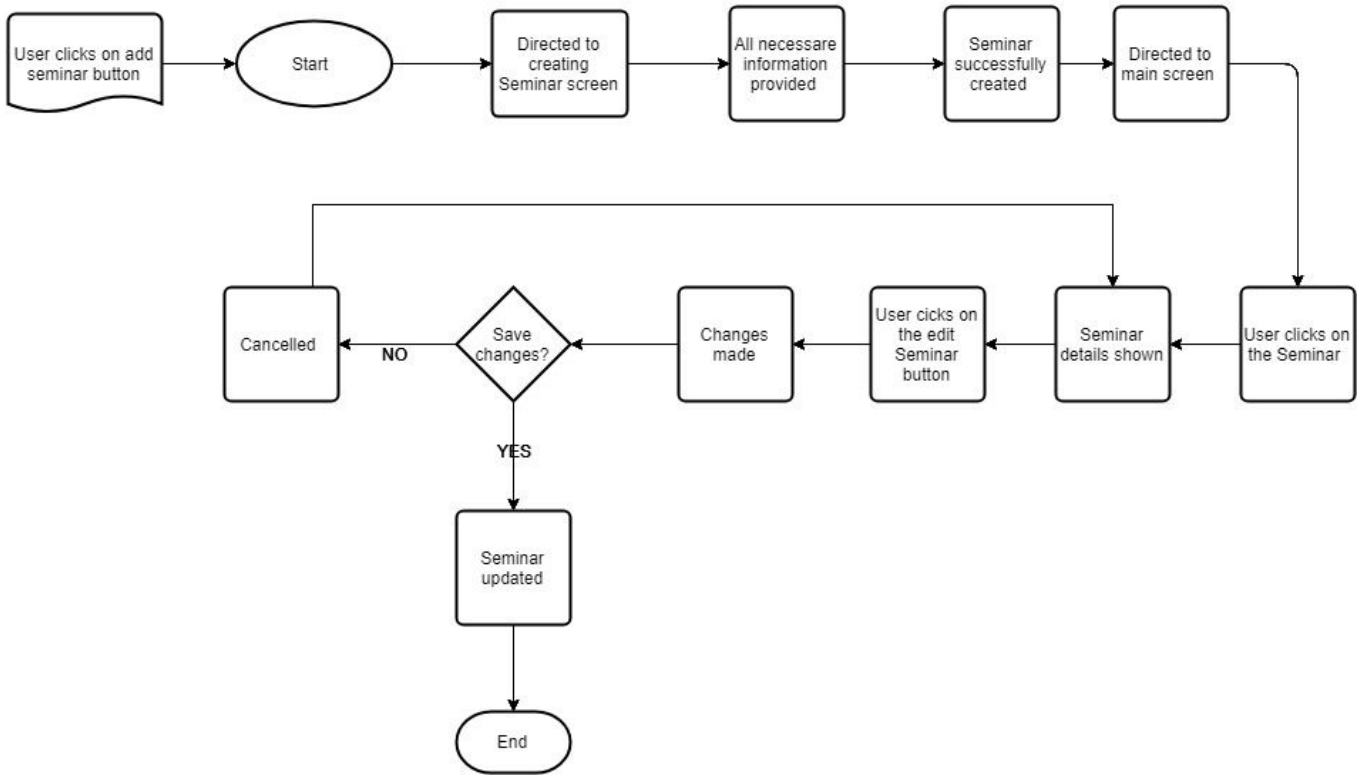
USER LOGIN



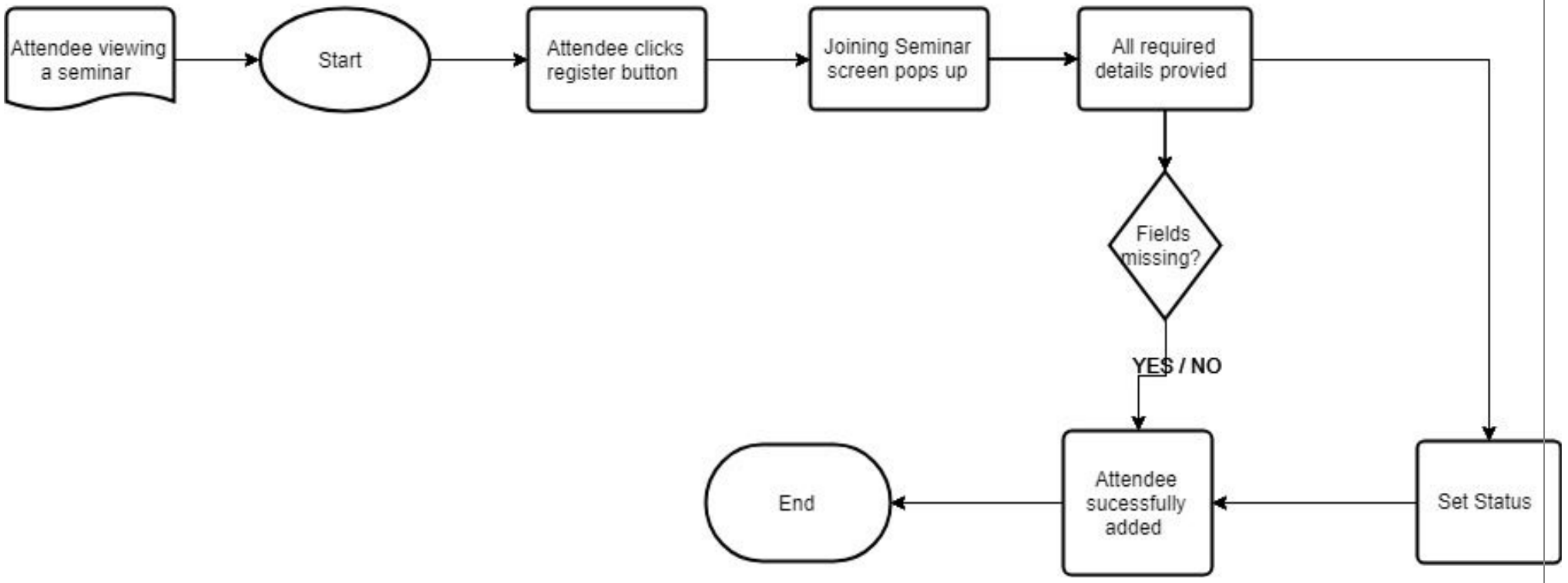
Register New User



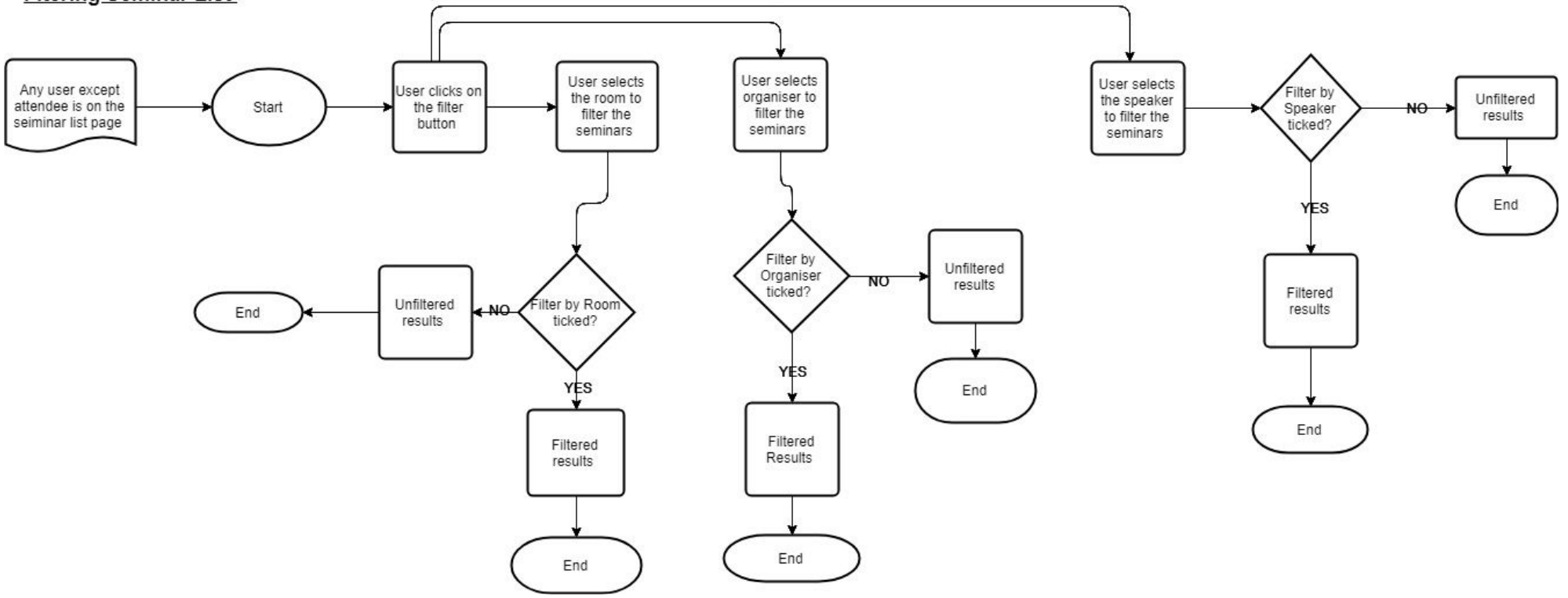
Create & Edit Seminar



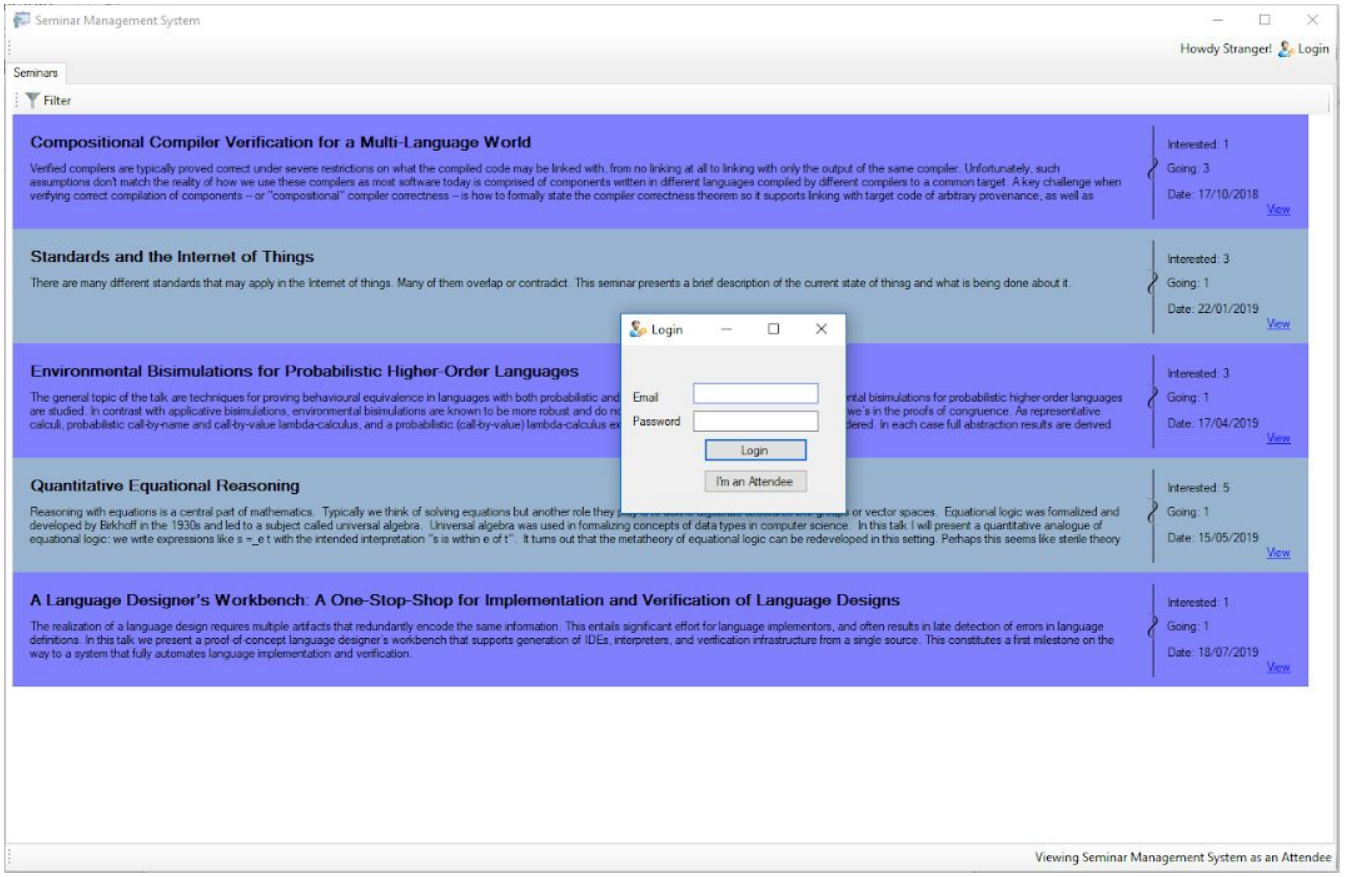
Attendee Joining a Seminar



Fitering Seminar List

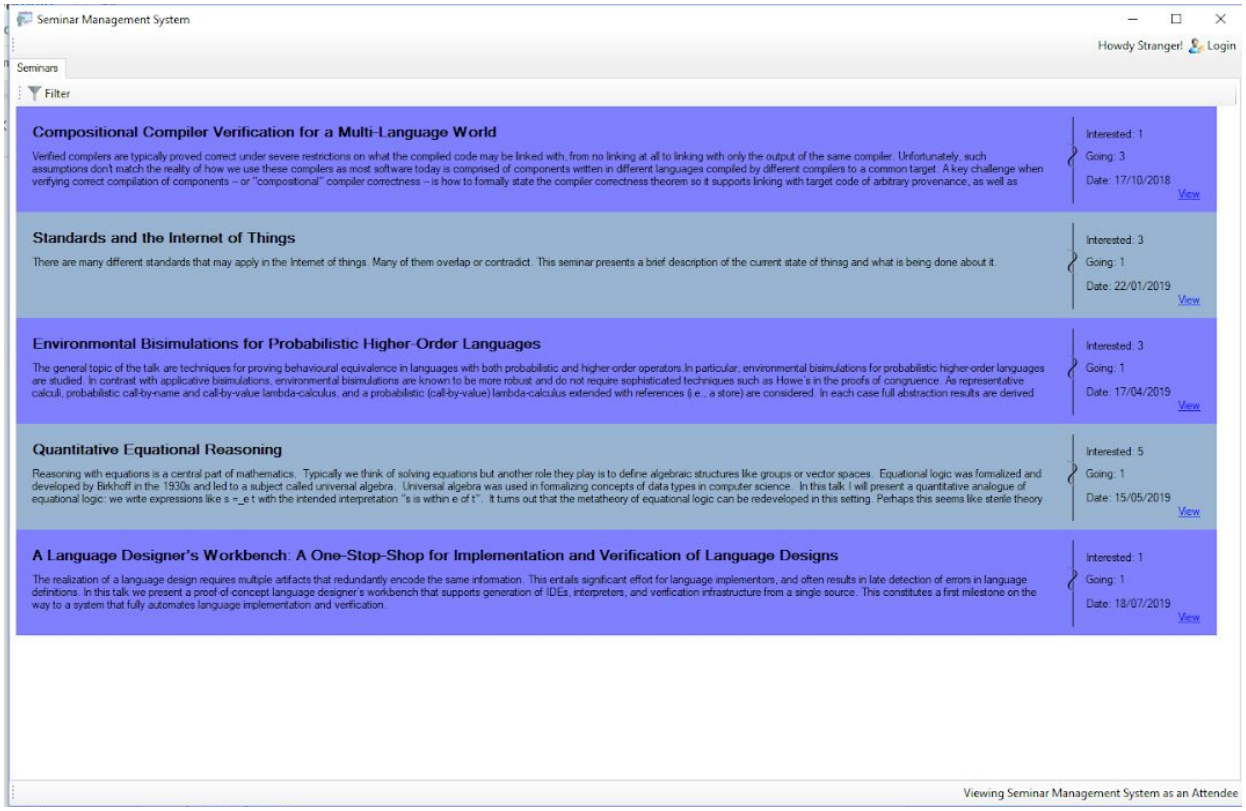


User Login



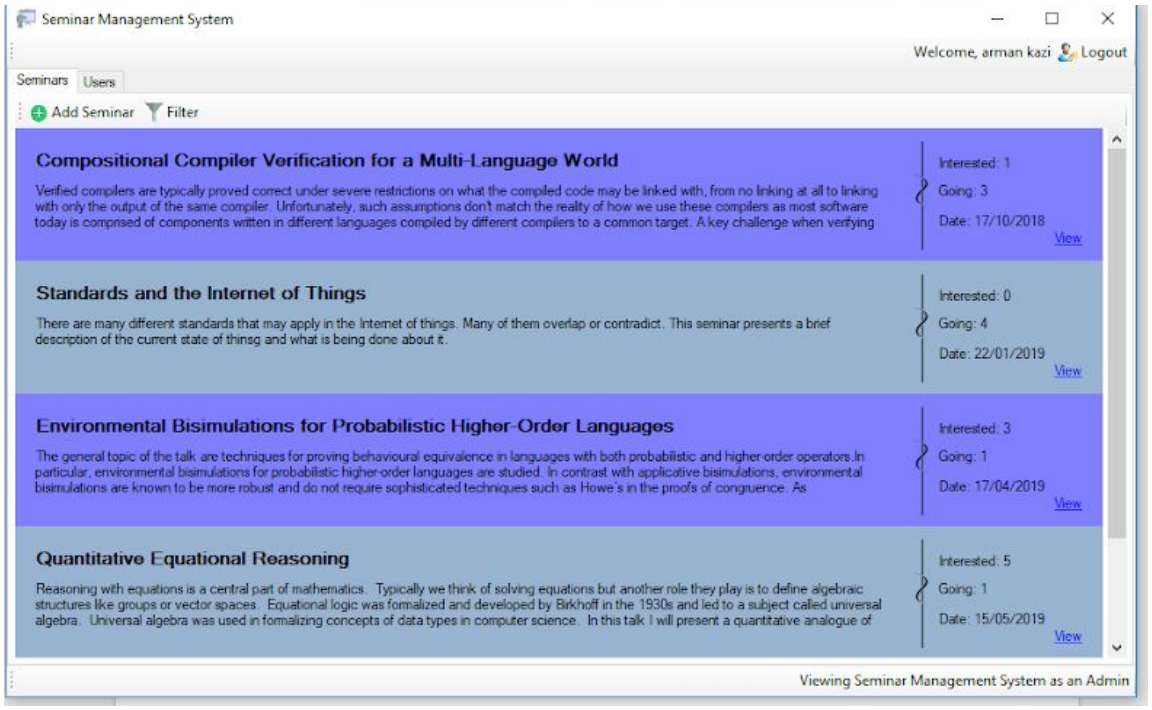
This is the user login page. We designed it to mirror how most programs design their login interfaces to ensure that users are able to login without any difficulty. If the user is an attendee, they can simply click on “I’m an Attendee” button and login. For any other users, they have to input their email to login. At this point in time, the implementation of password validation is out of scope and will be ignored.

Summary of Seminars

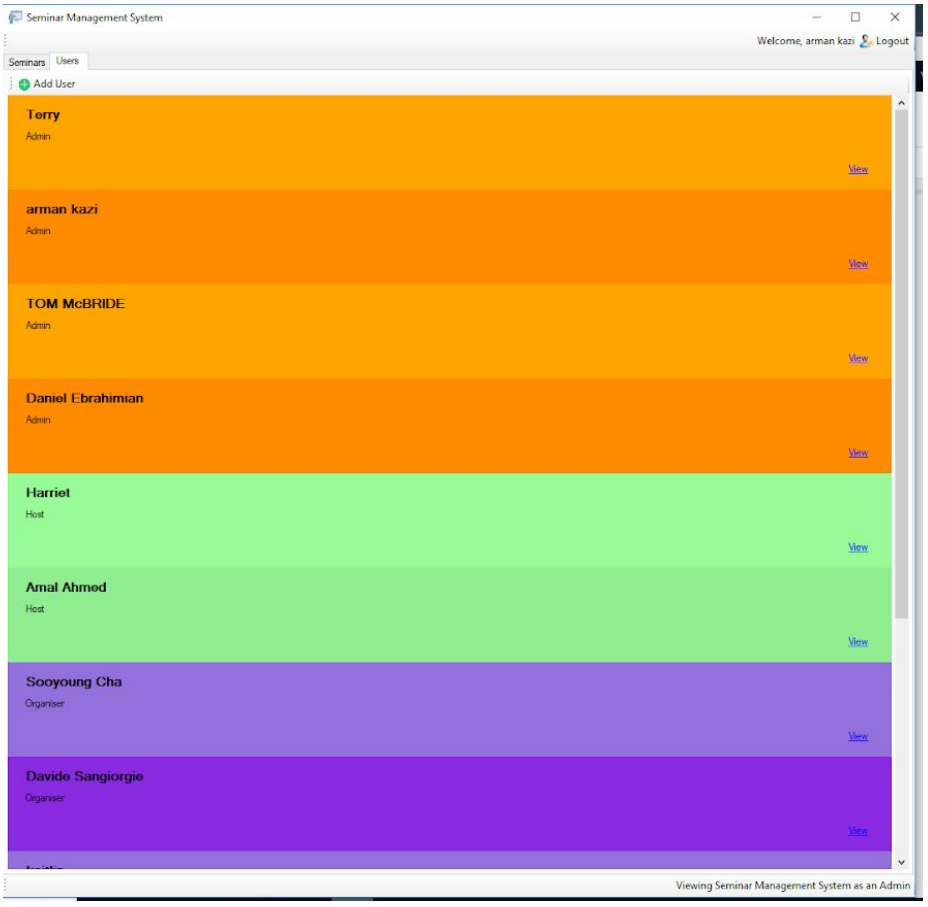


This the default screen of the system, the list of seminars page. Any user who logs in or starts the application will be directed to this page immediately. The current logged in user name will be displayed in the top right hand corner of the screen. This is an accessible way for users to know exactly who is using the system.

However, this screen is a little different to the admin and organizer's one as they have the authority to add seminars and users and view user list while others can not. This is shown below in the screenshot.

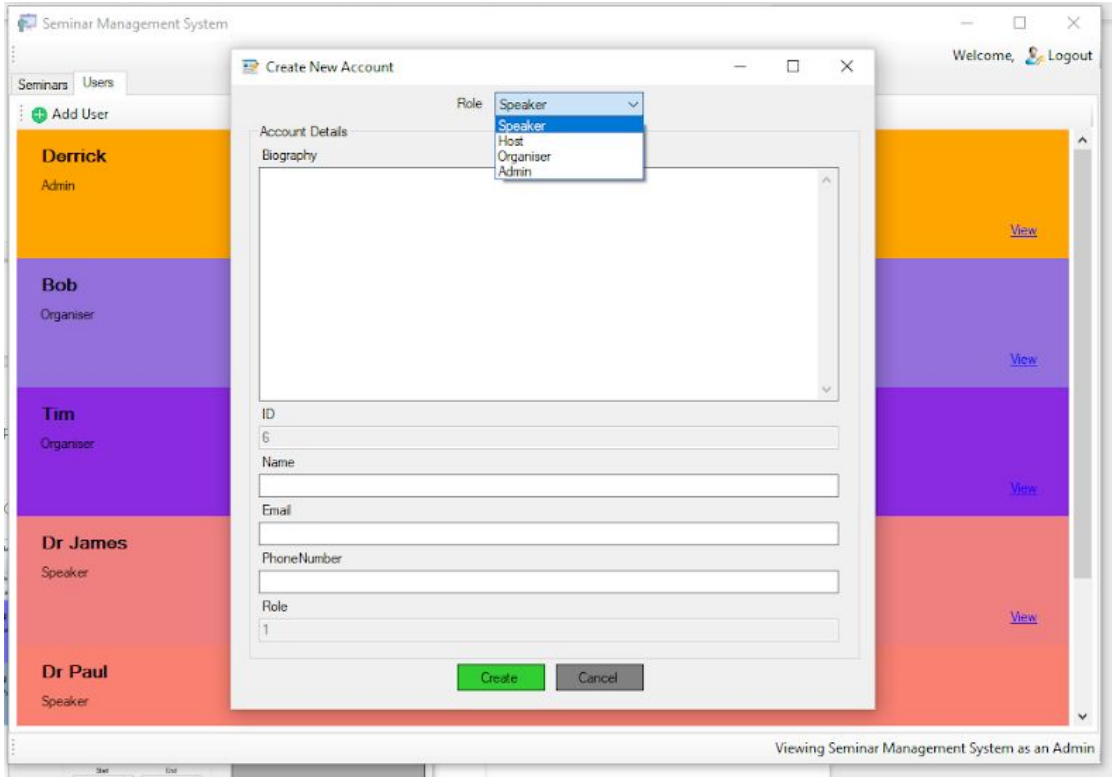

Seminar Management System
Welcome, arman kazi Logout
Seminars
Users
Add Seminar
Filter
Compositional Compiler Verification for a Multi-Language World
Verified compilers are typically proved correct under severe restrictions on what the compiled code may be linked with, from no linking at all to linking with only the output of the same compiler. Unfortunately, such assumptions don't match the reality of how we use these compilers as most software today is comprised of components written in different languages compiled by different compilers to a common target. A key challenge when verifying
Interested: 1
Going: 3
Date: 17/10/2018
View
Standards and the Internet of Things
There are many different standards that may apply in the Internet of things. Many of them overlap or contradict. This seminar presents a brief description of the current state of thinsg and what is being done about it.
Interested: 0
Going: 4
Date: 22/01/2019
View
Environmental Bisimulations for Probabilistic Higher-Order Languages
The general topic of the talk are techniques for proving behavioural equivalence in languages with both probabilistic and higher-order operators.In particular, environmental bisimulations for probabilistic higher-order languages are studied. In contrast with applicative bisimulations, environmental bisimulations are known to be more robust and do not require sophisticated techniques such as Howe's in the proofs of congruence. As
Interested: 3
Going: 1
Date: 17/04/2019
View
Quantitative Equational Reasoning
Reasoning with equations is a central part of mathematics. Typically we think of solving equations but another role they play is to define algebraic structures like groups or vector spaces. Equational logic was formalized and developed by Birkhoff in the 1930s and led to a subject called universal algebra. Universal algebra was used in formalizing concepts of data types in computer science. In this talk I will present a quantitative analogue of
Interested: 5
Going: 1
Date: 15/05/2019
View
Viewing Seminar Management System as an Admin

User List



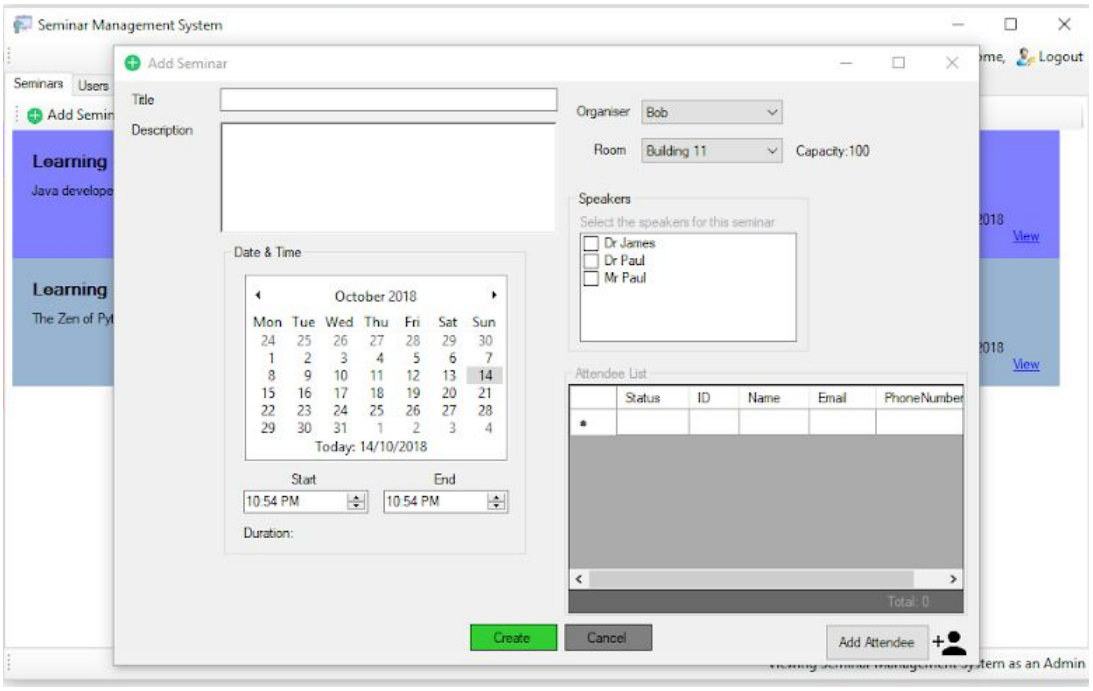
This is the user list. Only admins have the authority to view this page. They can change the details of any user, update their role or even delete them. The colour for different user is quite different. This assists the administrator by visually distinguishing different user types so that they can easily see what role each user has.

Register a new user



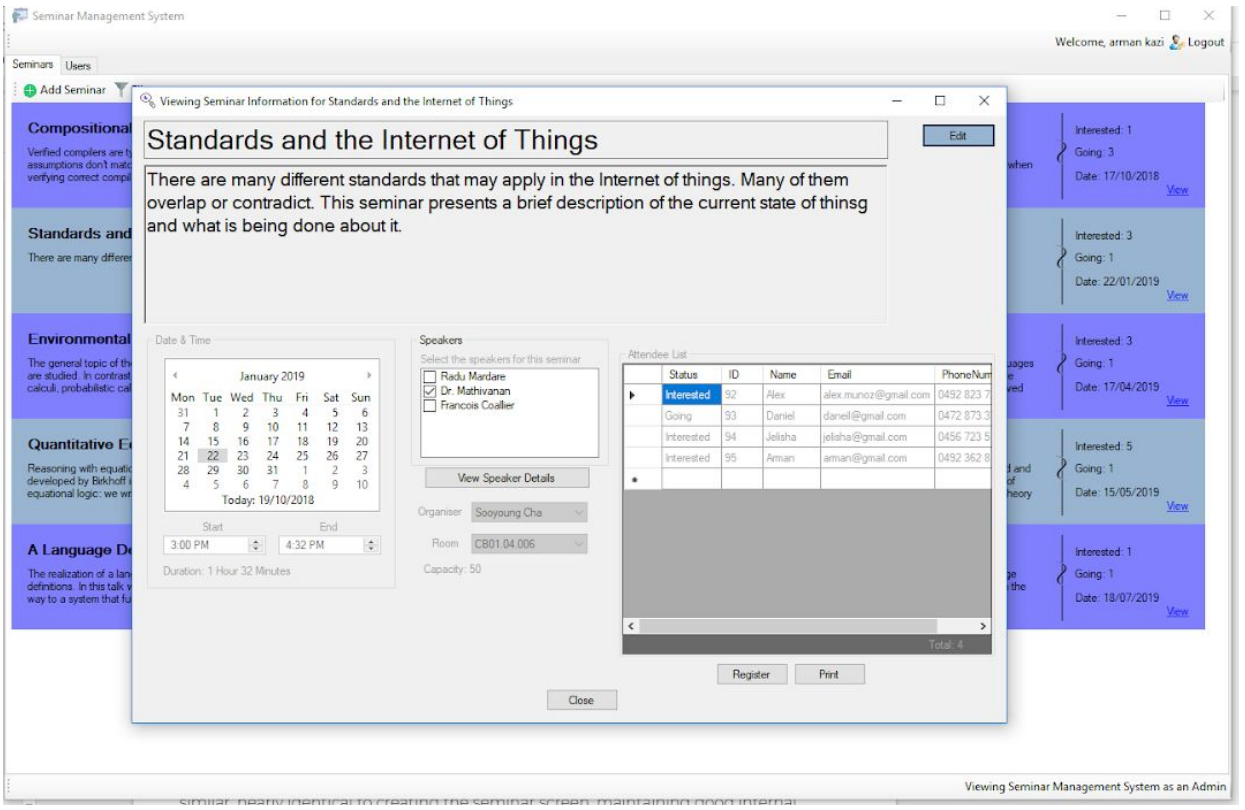
This screen is for registering new users. Only admins can view this page. Admins can create new users and assign them roles from a drop down list consisting of 4 options. In comparison to other users, only the speaker has the biography field as was specified by the client.

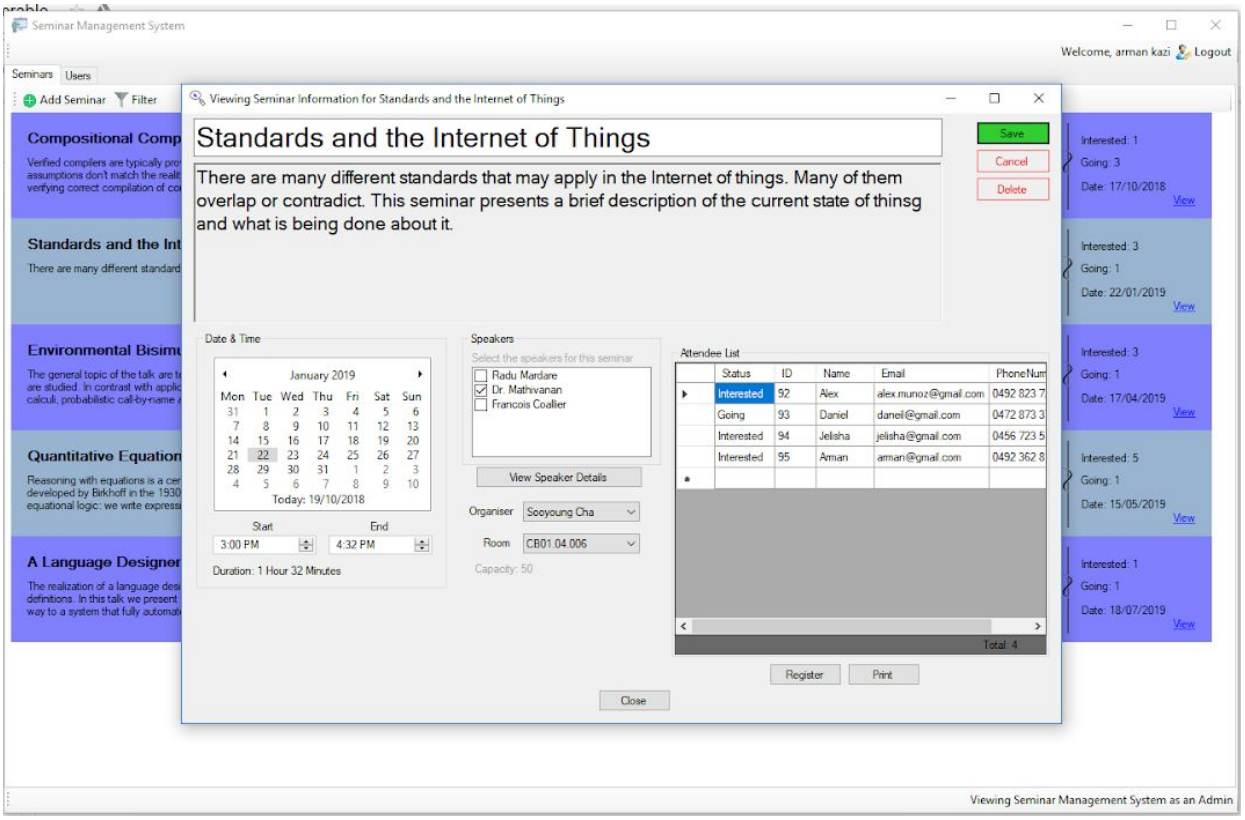
Create a Seminar



This page is for creating seminars. Only admins and organisers have the authority to create a seminar. The user needs to fill out all the information as prompted by the various fields. These include the drop down list for the venue, and selecting a date from the calendar. When choosing the date of seminar, the seminar will be set to invalid if the chosen date has passed. The user can also register attendees by simply clicking on “Add Attendee” button and filling out the details.

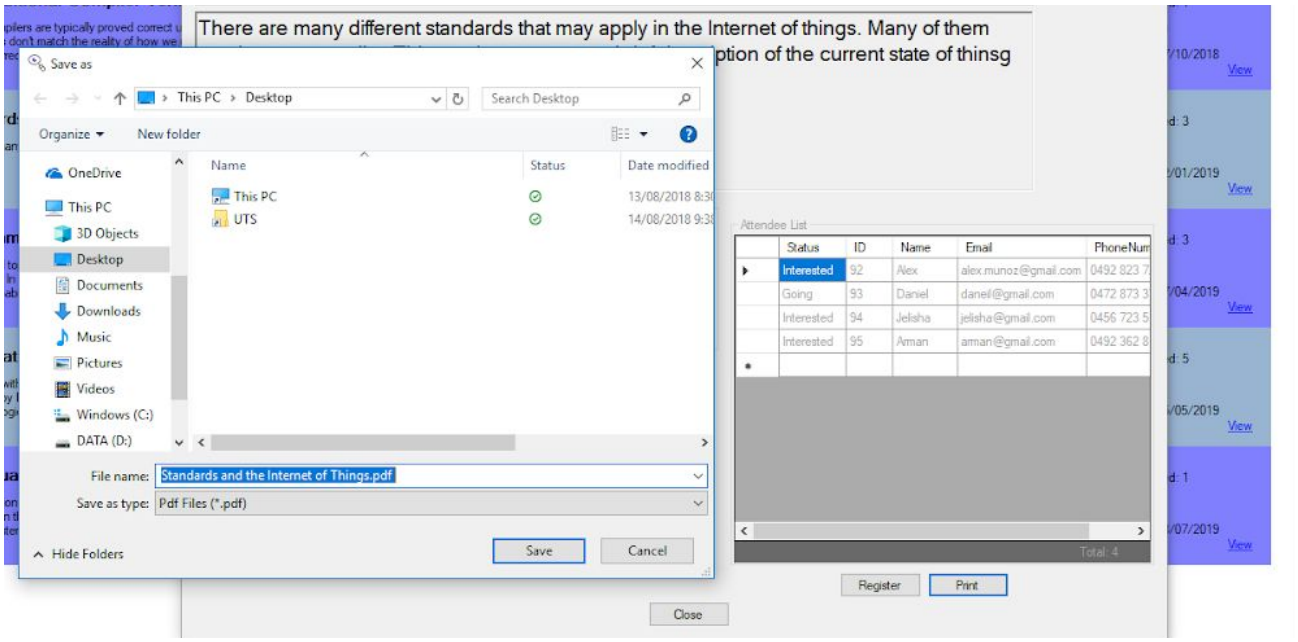
Edit a Seminar

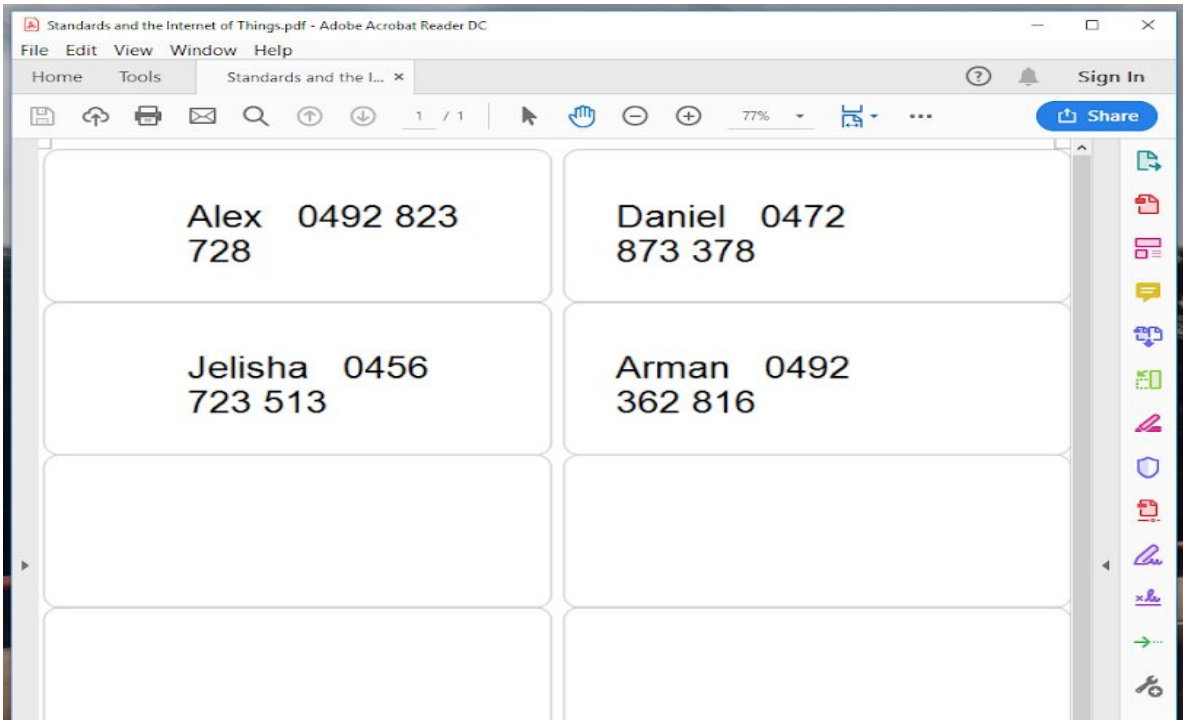




These screens show the process of editing the seminars. Once again only the admin and the organiser have the authority to access these screens. The layout of editing a seminar screen is similar, nearly identical to creating the seminar screen, maintaining good internal consistency within the system.

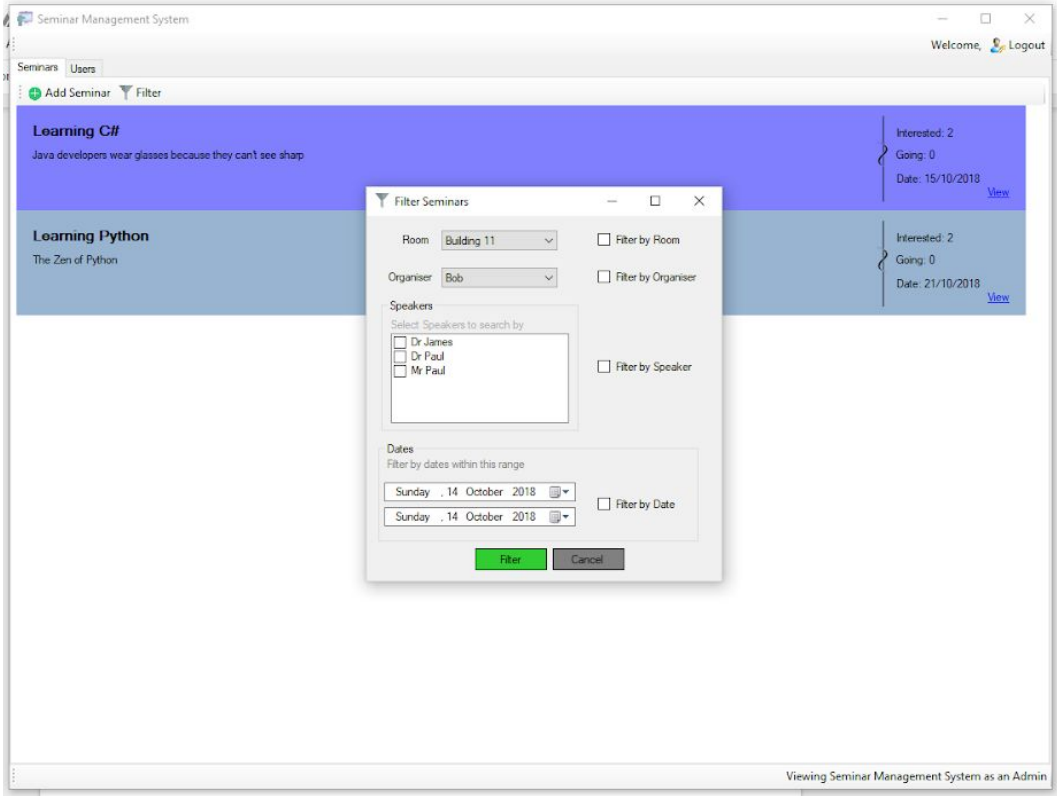
The print button creates a pdf as shown in the screenshots below with all the attendee names and a phone number to fit in a 14 slots per page Avery template.





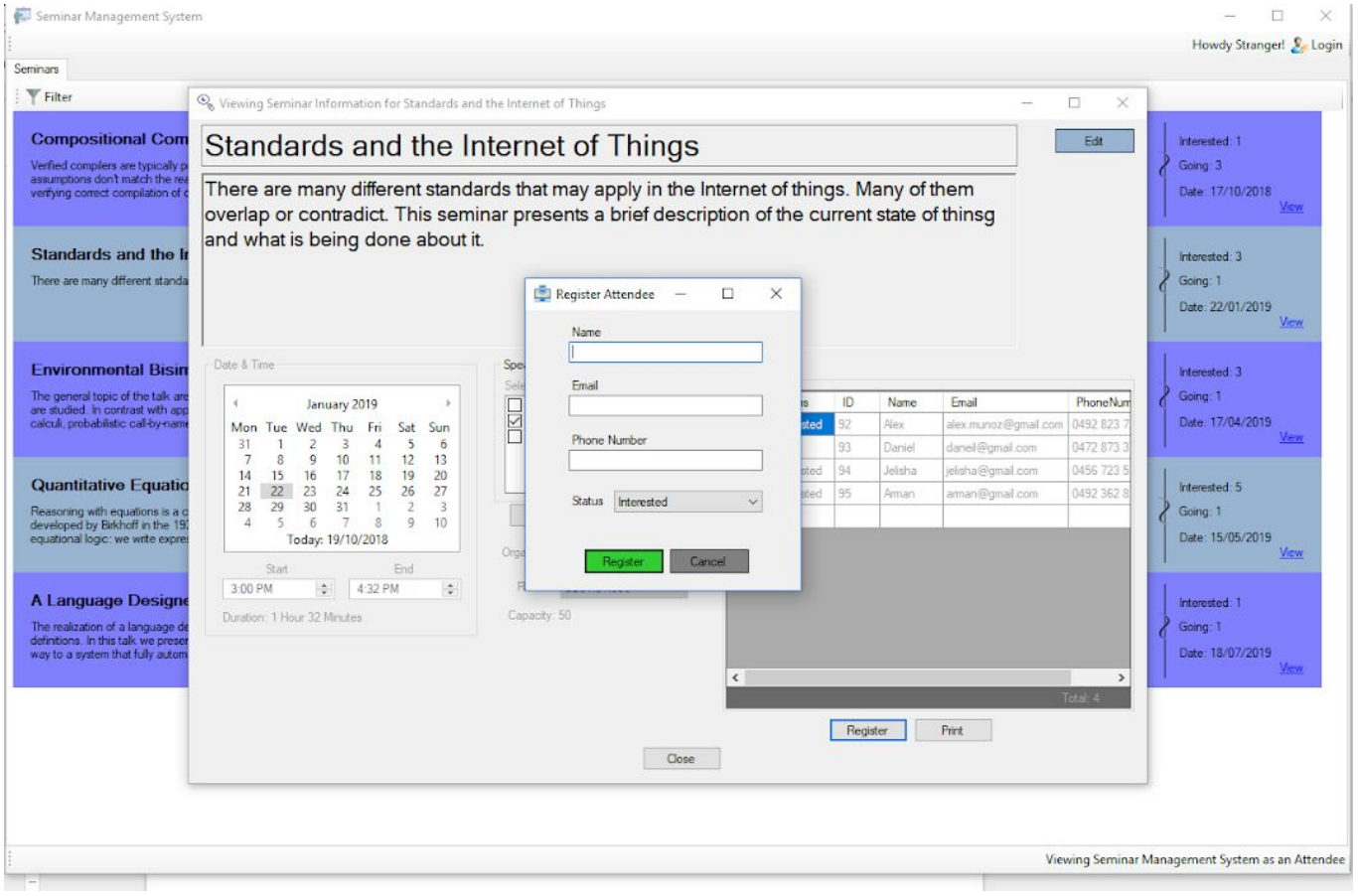
These screens show how the system prints the name tags of attendees to pdf. When viewing a seminar as an admin or organiser the print button in the lower right is available. Upon pressing the button if there are no attendees who are marked as “Going”, the name tags will not print. However, if there are attendees who are going, their names and numbers will be printed to a name tag which is housed in a pdf file. A save dialog will open and allow the user to choose where the pdf will be saved. After that point the pdf can be printed to cut up and distribute to the attendees.

Filter Seminars



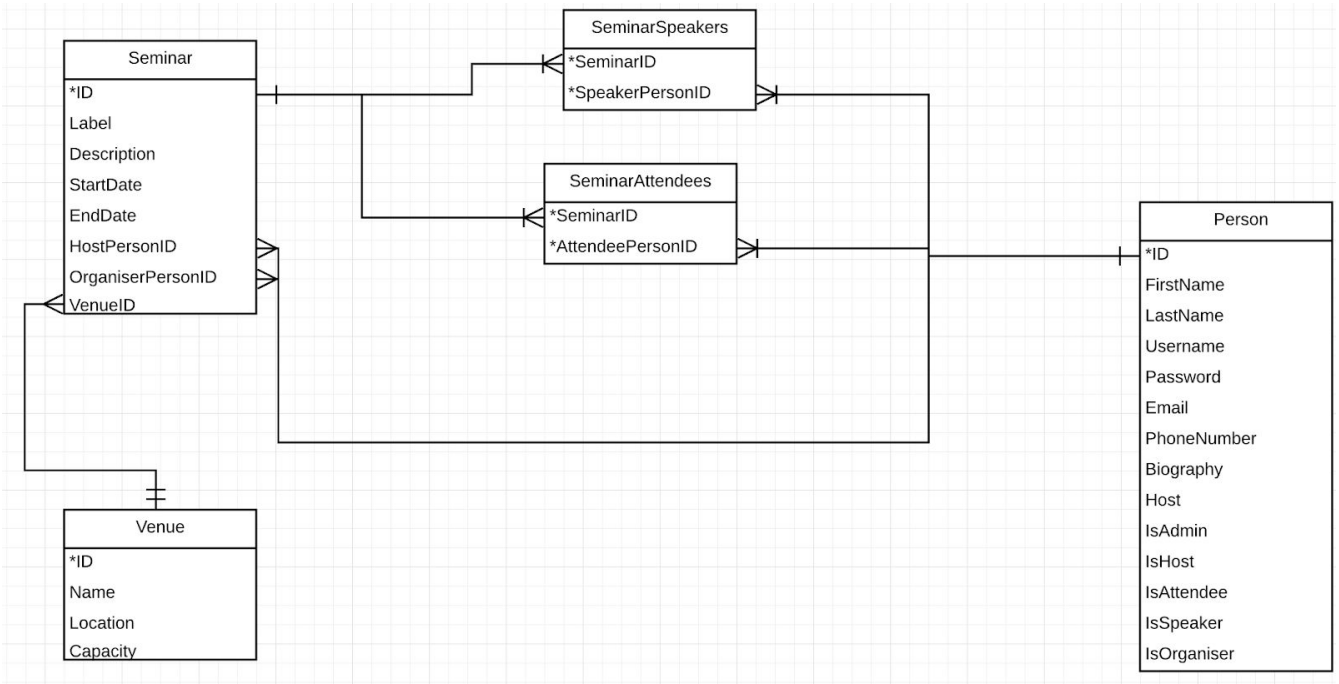
Any user can filter the seminars in 4 ways, either by room, organizer, speaker or by date. Room and Organiser needs to selected from a drop down list, the speakers are selected from a checklist and date can either be edited or can be selected from the drop down calendar option. When the user made a selection, they need to tick what they chose to filter the results with.

Attendee joining a seminar



This screen comes to play when an attendee views a seminar. They can register themselves simply by clicking on “Register” button and fill out the necessary information. It is up to the attendee if they want to provide their phone number if they wish to be contacted. The status is selected from a drop down list with the options of interested and going.

ERD DIAGRAM



We designed the database in a way that allows for one person to have multiple roles in a seminar. For example a speaker for a seminar can also be an organiser or an attendee. One seminar can have multiple speakers and attendees but can only have one host, organiser and venue.

The database provides us a large, secure and global space to store the data of our system. We assume that there are mountains of users from UTS to use the system. We need to store and handle the potential large chunks of data. Rather than a spreadsheet, having a database to store the data can reduce the time to get responding from the system. It supports much more users to manage data at one time without losing efficiency. Due to there are some personal information need to be stored, having a database is also convenient to secure those data. We can have security groups and privileges to restrict the unauthorized access. In our seminar management system, what we want most from having a database is that we can share the data globally among the user so that our application can work across many systems. Different users can access to the database and use the data in their seminar management system. As a result, user can use the system with the active data which is always up to date.

Data Table

We are using a data table to compliment the ERD and to give all developers a comprehensive understanding of the format and the user of the data that we are handling. It also provides the constraints we have set and gives examples of the data.

Ref	Data Element	Format	Length	Description	Example
01 - Person					
DE-1001	ID	Integer	8	The unique identifier for the user in the database	23982275
DE-1002	FirstName	Text	20	First name for the user	John
DE-1003	LastName	Text	20	Last name for the user	Smith
DE-1004	Username	Text	20	Users username	johniscool
DE-1005	Password	Text	20	Users password	johnisthebest
DE-1006	Email	Text	50	Users email address	john@student.uts.edu.au
DE-1007	Phone Number	Text	10	Users phone number	0476 876 293
DE-1008	Biography	Text	200	A biography that gives users insight to the speakers life	Hello this is my biography
DE-1009	Host	Text	50	The name of the Host that is sponsoring the	FEIT

				event	
DE-1010	IsAdmin	Boolean		Boolean that determines if this user is an admin	True
DE-1011	IsHost	Boolean		Boolean that determines if this user is a host	False
DE-1012	IsAttendee	Boolean		Boolean that determines if this user is a attendee	True
DE-1013	IsSpeaker	Boolean		Boolean that determines if this user is a speaker	False
DE-1014	IsOrganiser	Boolean		Boolean that determines if this user is a organiser	True
DE-1015	IsUser	Boolean		Boolean that determines if this user is a user	False
02 - Seminar					
DE-2001	ID	Integer	8	Unique Identifier for seminar	5
DE-2002	Label	Text	30	Title of the seminar	Python Basics
DE-2003	Description	Text	200	A description of the seminar	Seminar to learn the basics of python
DE-2004	StartDate	DateTime		The starting time of the seminar	20-JUN-2018 08:00:00

DE-2005	EndDate	DateTime		The end time of the seminar	20-JUN-2018 010:00:00
DE-2006	HostPersonID	Integer	8	The host persons ID for this seminar	8
DE-2007	OrganiserPersonID	Integer	8	the organiser persons ID for this seminar	24
DE-2008	VenueID	Integer	8	The venue ID for this seminar	13
03 - Venue					
DE-3001	ID	Integer	8	The unique identifier for the venue	12
DE-3002	Location	Text	50	The location of the venue	B11.05.200
DE-3003	Capacity	Integer	10	The amount of people that can fit in the venue	50

Coding Style

When creating scalable software solutions, it is important to maintain the readability and design of the code base. This can be achieved by agreeing on a standard style guide to use within the project. By utilizing a style guide we can assume a uniform consistency across files, regardless of which developer created it.

Having a central style guide means that all developers working on the project must follow the coding style to ensure consistent and readable code. It makes it so that other developers are easily able to follow code that has been written by other developers.

The style guide outlines coding conventions that when followed, ensures readable code. These conventions include proper line breaks, meaningful naming scheme, code reusability and clear comments. As there are some members of the group who do not have much experience with coding in C#, commenting code is vital for understanding the purpose of the code. Comments in the code need to express what the intended purpose is of the code. This enables greater collaboration between team members and system maintainability for future developers.

Style guides also assist the developer in making decisions about how to structure a statement. It reduces the cognitive overhead of trying to structure a complex statement. These statement formats are defined in the style guides and can be referred to at any moment.

This project is created with C#. Therefore we must identify a style guide that was created for this language. Finding a suitable style guide for this language is not a straightforward as compared to other languages such as Python, which can always refer to the standard style guide outlined in PEP-08 (Python Enhancement Proposal 8). Rather, there is a large selection to choose from or combined guides made by community members. Some of these style guides have strong opinions about introducing new and different ways of expressing code. The guides in this style didn't work well with our development team as it introduced too many new style changes to what our team has used in the past.

Our selection for a style guide was narrowed down to two main options. The first being published by dofactory (Dofactory - C# Coding Standards) and the second published by Microsoft (Microsoft - Coding Conventions). There are large differences between the two options in terms of how the rules and conventions are visually depicted. Dofactory represents the rules with the use of spacing, bolding and coloring. Each section is clearly defined with a "do" or "do not". Whereas Microsoft's guide uses traditional bullet points and headings. The style guide becomes an information overload and makes navigating hard because sections

don't stand out. Dofactory provides explicit examples of what is correct, what should be avoided and what are exceptions to style rules. This level of detail wasn't as obvious in Microsoft's documentation. We found the ease of use provided by dofactory to be beneficial to our team and decided to use that as our style guide.

SAMPLE SNIPPETS FROM DOFACTORY GUIDE

do declare all member variables at the top of a class, with static variables at the very top.

```
// Correct
public class Account
{
    public static string BankName;
    public static decimal Reserves;

    public string Number {get; set;}
    public DateTime DateOpened {get; set;}
    public DateTime DateClosed {get; set;}
    public decimal Balance {get; set;}

    // Constructor
    public Account()
    {
        // ...
    }
}
```

Why: generally accepted practice that prevents the need to hunt for variable declarations.

do not use **Underscores** in identifiers. Exception: you can prefix private static variables with an underscore.

```
// Correct
public DateTime clientAppointment;
public TimeSpan timeLeft;

// Avoid
public DateTime client_Appointment;
public TimeSpan time_Left;

// Exception
private DateTime _registrationDate;
```

Why: consistent with the Microsoft's .NET Framework and makes code more natural to read (without 'slur'). Also avoids underline stress (inability to see underline).

avoid using **Abbreviations**. Exceptions: abbreviations commonly used as names, such as **Id, Xml, Ftp, Uri**

```
// Correct
UserGroup userGroup;
Assignment employeeAssignment;

// Avoid
UserGroup usrGrp;
Assignment empAssignment;

// Exceptions
CustomerId customerId;
XmlDocument xmlDocument;
FtpHelper ftpHelper;
UriPart uriPart;
```

Why: consistent with the Microsoft's .NET Framework and prevents inconsistent abbreviations.

DEVELOPMENT ENVIRONMENT

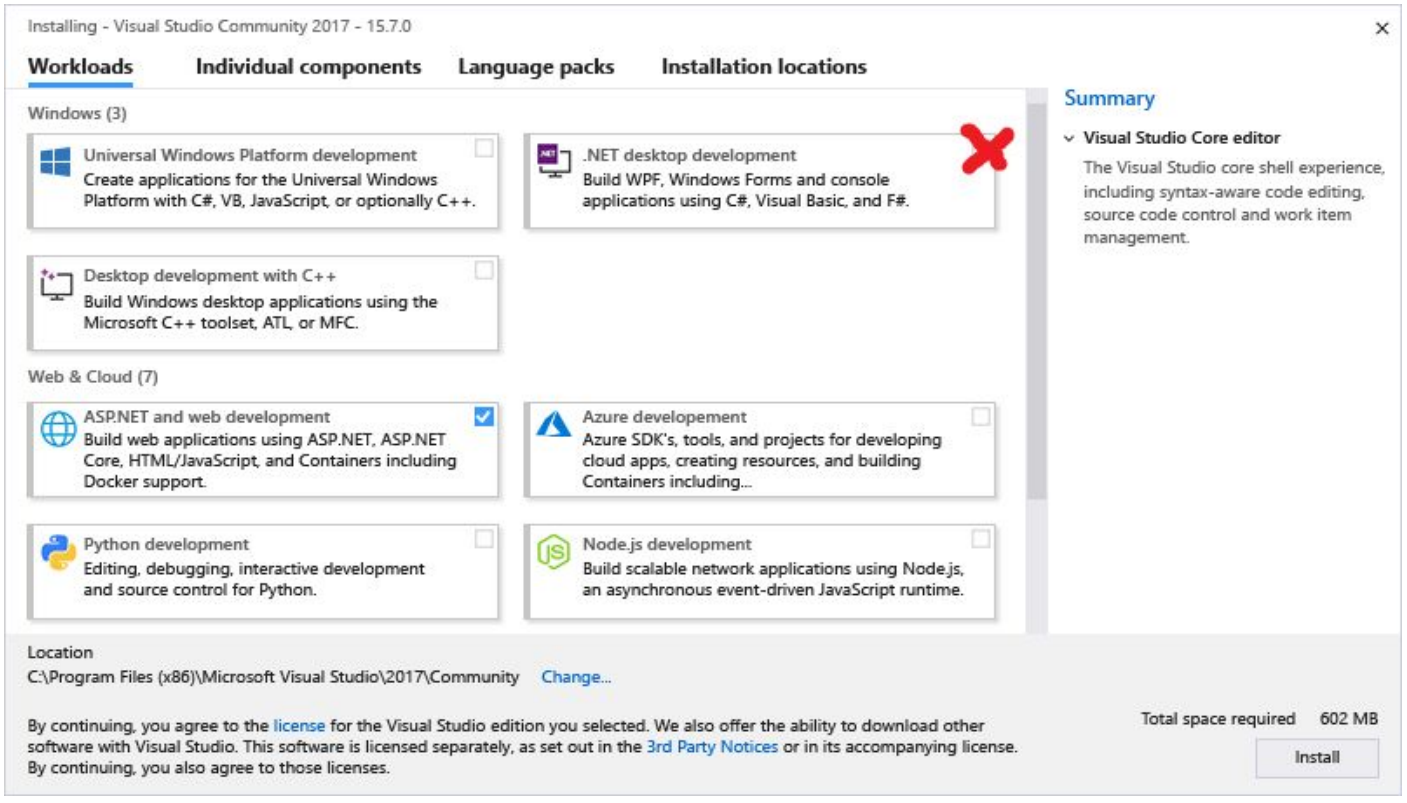
The software for this Seminar Management System was made with Visual Studios Community. Visual Studios is an Integrated Development Environment (IDE), primarily used to develop Windows Applications. We utilised this IDE because it is a matured piece of Software that is constantly receiving updates from Microsoft.

INSTALLATION

INSTALL VISUAL STUDIO COMMUNITY

Visual Studios Community edition can downloaded from [Download Visual Studio Community]

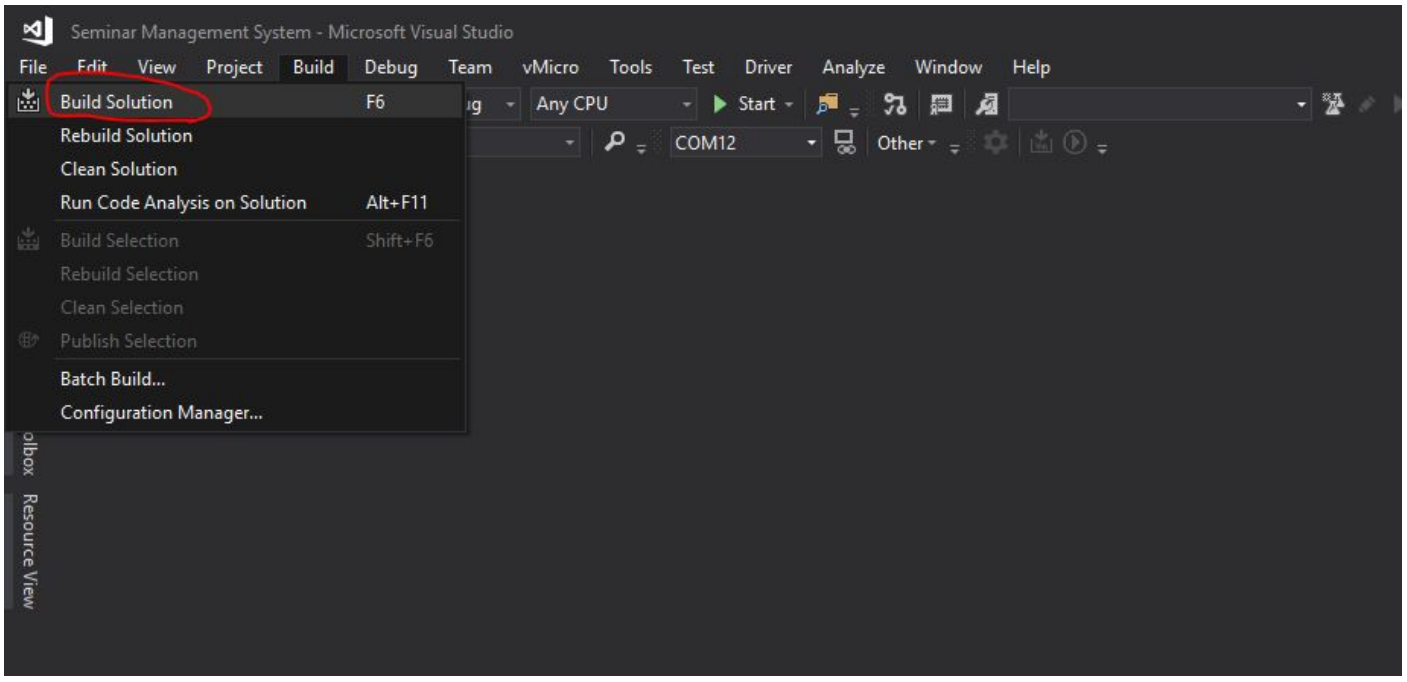
When installing the software, make sure to select the ".NET Desktop Development" workload as shown below:



Once Visual Studios Community has been installed, the project can now be loaded.

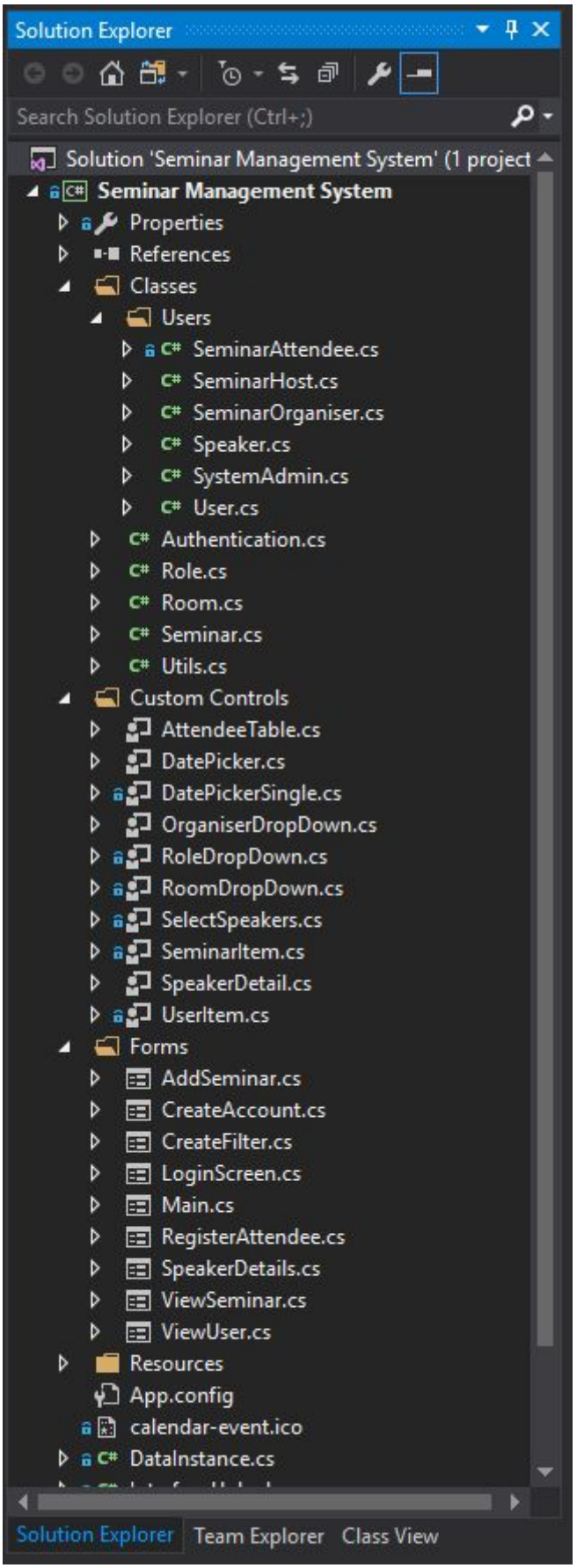
Loading the solution

The source code of the project will be provided alongside this report. To load the solution into Visual Studios correctly it is recommended to navigate to the folder 'src/Seminar Management System/Seminar Management System/' and then open the 'Seminar Management System.csproj' file with Visual Studios. Once it has loaded, go to 'Build' and click 'Build Solution' as show in the figure below.



This will build all the local dependencies required and compile an executable binary file.

At this point, the software should be in a state where it is editable. The following figure is a sample of what will be visible in the 'Solution Explorer' in Visual Studios.

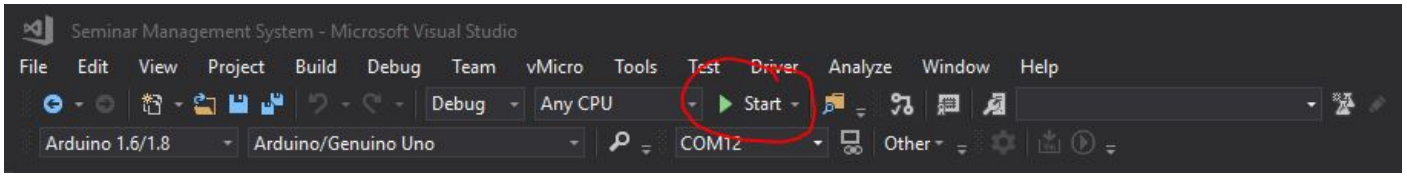

Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Seminar Management System' (1 project
Seminar Management System
Properties
References
Classes
Users
SeminarAttendee.cs
SeminarHost.cs
SeminarOrganiser.cs
Speaker.cs
SystemAdmin.cs
User.cs
Authentication.cs
Role.cs
Room.cs
Seminar.cs
Utils.cs
Custom Controls
AttendeeTable.cs
DatePicker.cs
DatePickerSingle.cs
OrganiserDropDown.cs
RoleDropDown.cs
RoomDropDown.cs
SelectSpeakers.cs
SeminarItem.cs
SpeakerDetail.cs
UserItem.cs
Forms
AddSeminar.cs
CreateAccount.cs
CreateFilter.cs
LoginScreen.cs
Main.cs
RegisterAttendee.cs
SpeakerDetails.cs
ViewSeminar.cs
ViewUser.cs
Resources
App.config
calendar-event.ico
DataInstance.cs
Solution Explorer
Team Explorer
Class View

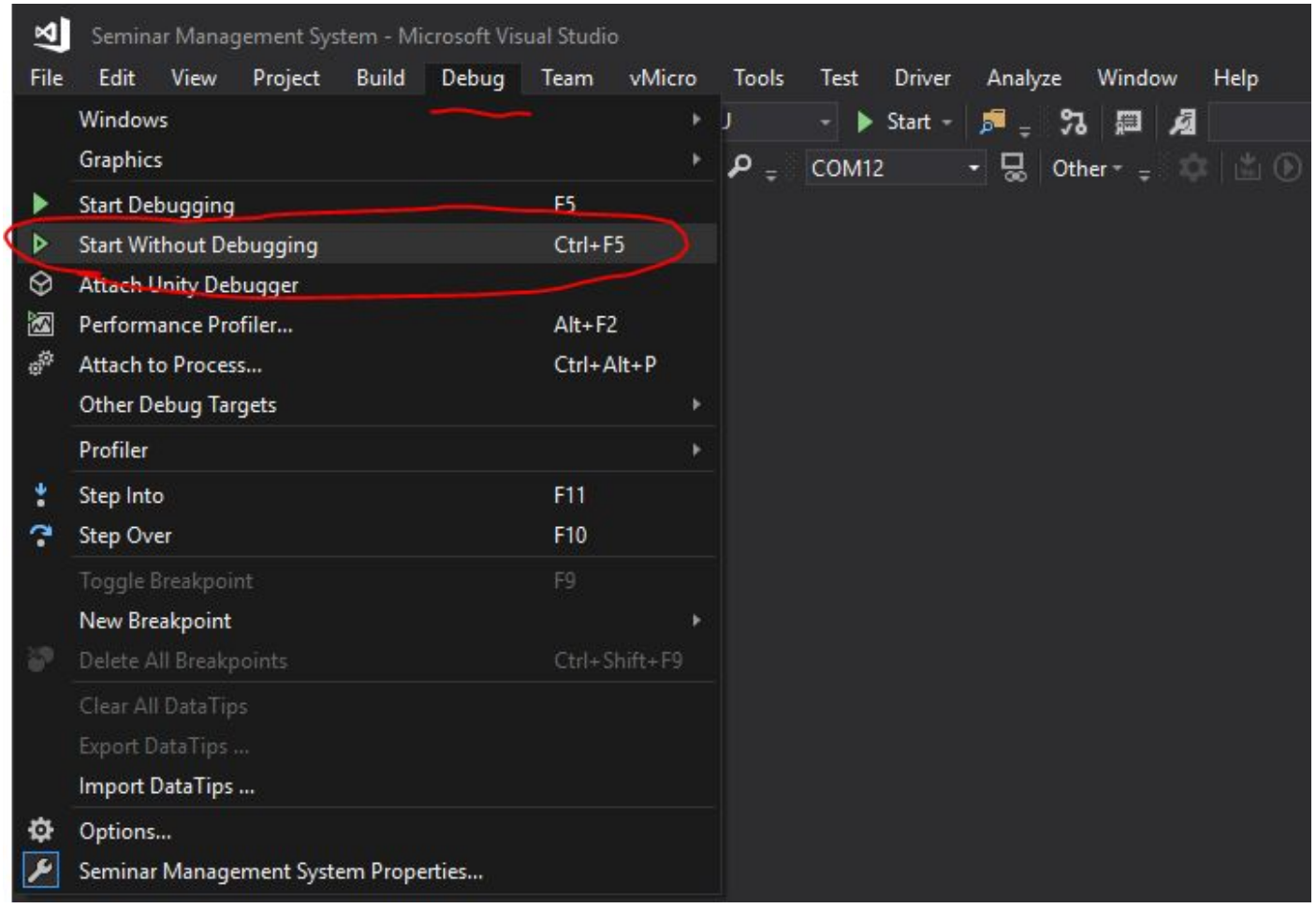
Running the solution

The solution can be ran from two primary locations. One within Visual Studios and one externally. When executing within Visual Studios, certain features are gained such as insights into resource usage and garbage collection. Executing externally is a similar experience to executing any normal program such as Google Chrome.

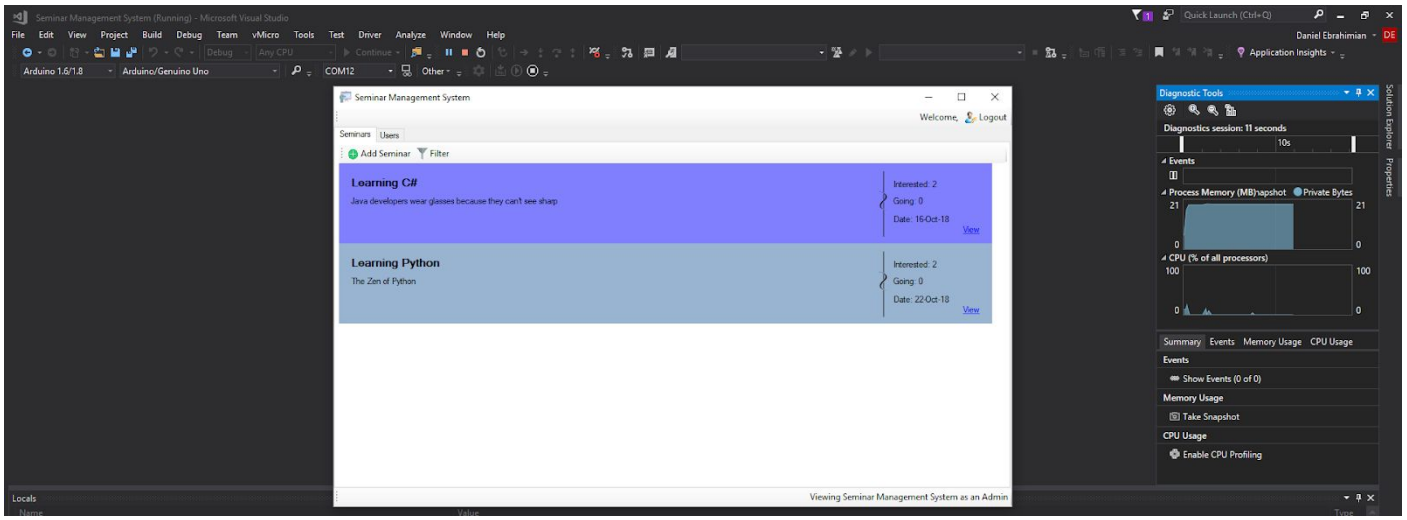
Executing within Visual Studios

Open the solution in Visual Studios. There should be a large 'Start' button located on the top tool strip will be a large. Alternatively, this can be done from the 'Debug' menu strip and clicking on 'Start Debugging' or by pressing CTRL + F5.





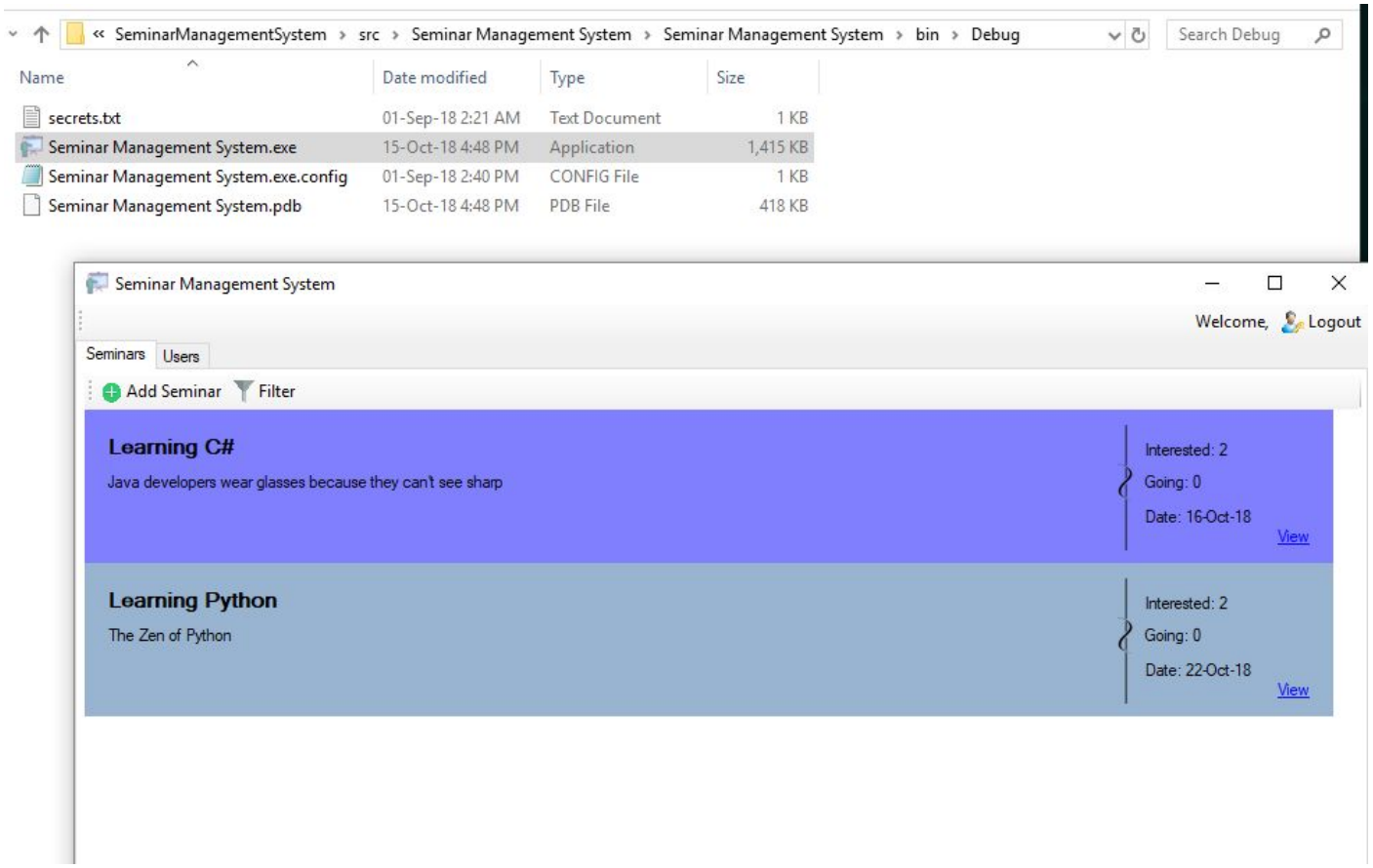
This program will then appear as such:



EXECUTING EXTERNALLY

When the build process completed in Visual Studios, a binary is created in the 'bin' folder of the project. This folder can be found in 'Seminar Management System/src/Seminar Management System/Seminar Management System/bin/debug' the file will be named 'Seminar Management System.exe'.

Opening this file will launch the software.



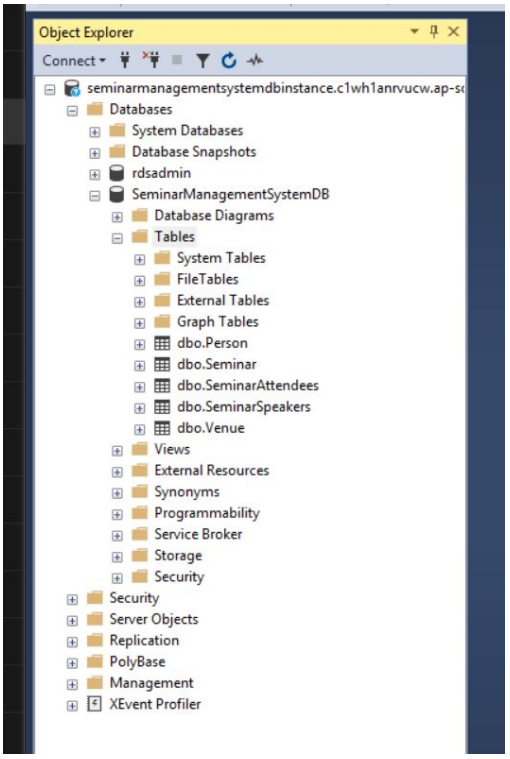
Database Environment

The database environment we have chosen for this software is Microsoft SQL server express. To manipulate the database we must install a program called Microsoft SQL Server Management Studio.

Setting up the environment:

- Download and install SQL Server Management Studio
- Connect to new server
- Server name: seminarmanagementsystemdbinstance.c1wh1anrvucw.ap-southeast-2.rds.amazonaws.com
- Click connect

After connecting to the server, the tables of the database can be accessed.



There are 5 tables that we are working with. These tables are as follows:

- Person table: this table stores all information regarding users, user information and their roles
- Seminar table: this table holds all information about the seminar and what users are involved in creating the seminar
- Seminar Attendees table: this table has all the attendees linked to the seminars that they are attending
- Seminar Speakers: this table has all the speakers linked to the seminars that they are speaking at
- Venue: this has all the venues stored in it

Using Microsoft Sql Server Management Studio is quite simple, to view a tables data you can right click on it then click “Select top 1000 rows”, this automatically generates a SQL script that has a select statement in it.

To create a new SQL script just press “Ctrl + N” on your keyboard, and the shortcut to execute this script is “F5” on the keyboard.

Source Control

The source code for this software was managed with Github. Access to this repository won't be provided with this report, however this can be requested. The following will be instructions on how to connect the source code to a new repository.

Begin by opening a terminal session with git installed and the source code folder as the working directory in the terminal. Ensure this folder is the root of the solution and not a sub-folder. For example:

```
$ cd ~/Seminar Management System
```

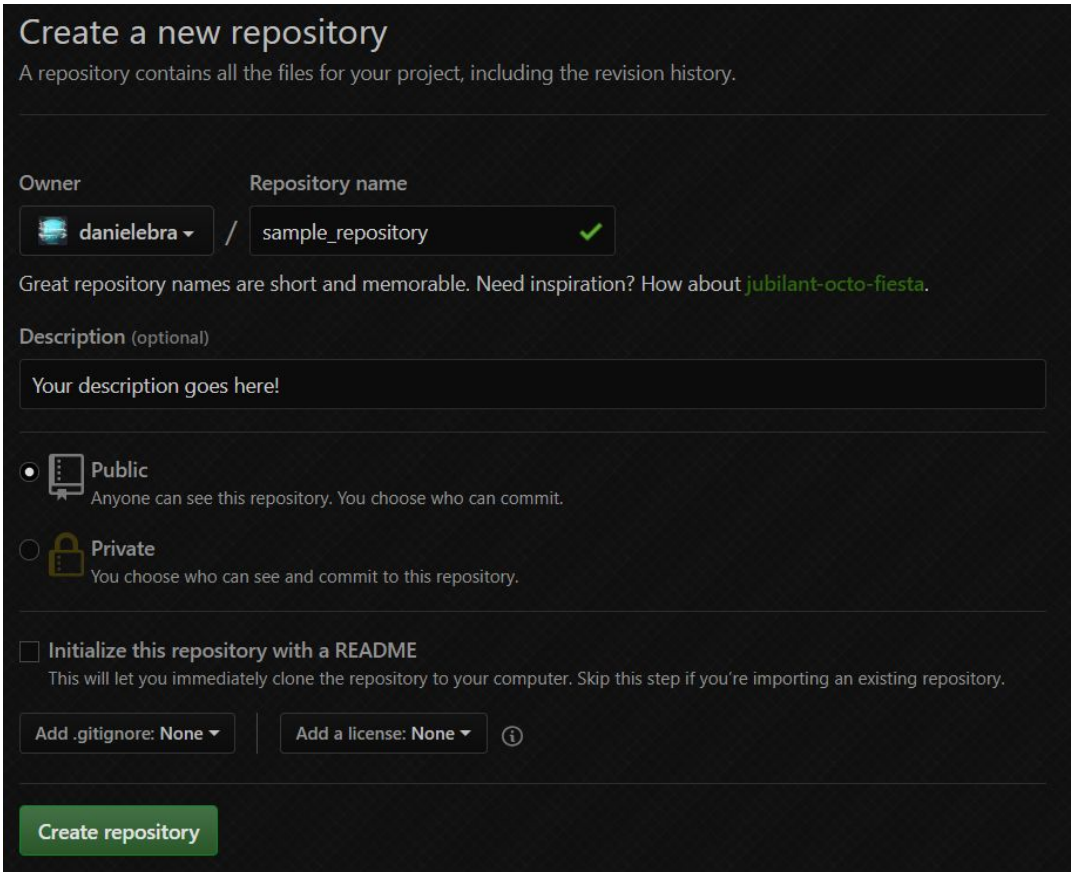
Once in the correct directory, git can be initialised with:

```
$ git init
```

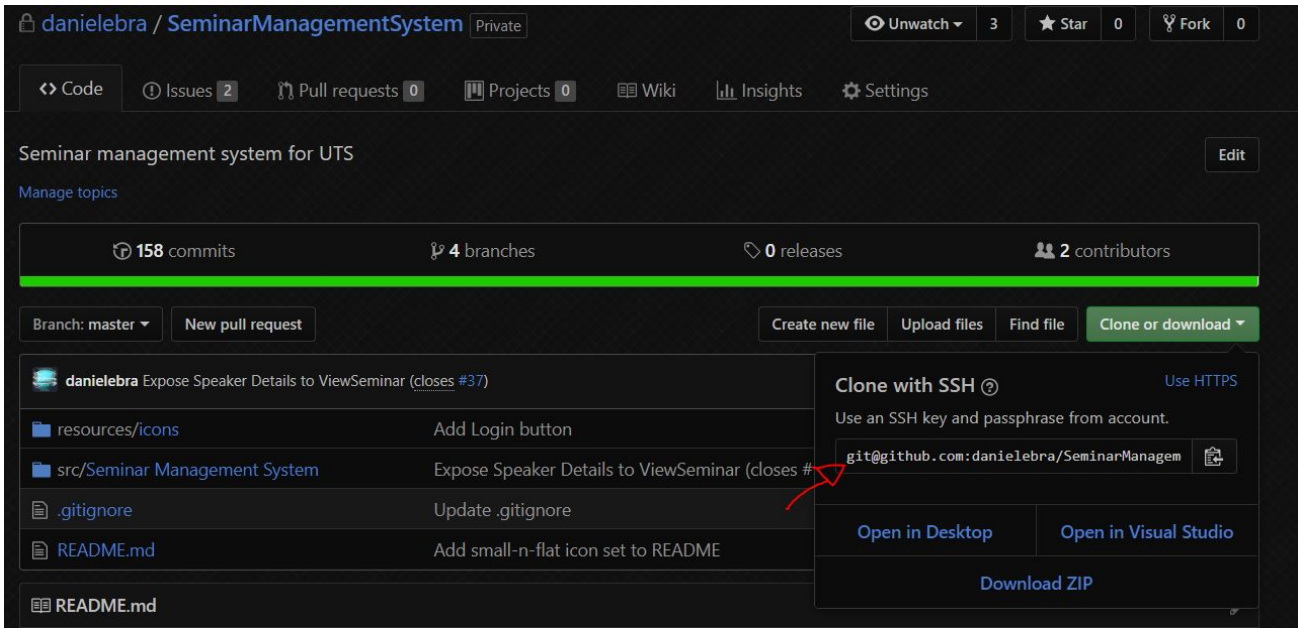
In order to connect this local repository to one that is hosted remotely, a remote path is needed. For this guide, Github will be used.

Navigate to https://github.com/new to create a new repository (create an account if needed)

Fill in the relevant information such as 'Repository name' and 'Description' as shown in the figure below.



By creating a repository a link will be provided that can be used as the remote path. This can be accessed via the ‘Clone or download’ button, as seen in the figure below.



Now return to the terminal and complete the following commands:

```
$ git remote add origin <link from ‘Clone or download’ button>
```

Normal git commands will now communicate with the online repository, such as:

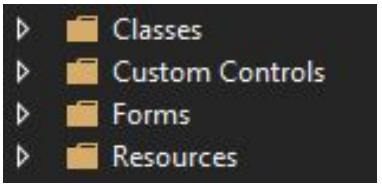
```
$ git pull origin master
```

Implementation

The software implemented in this system has been developed with the Model-View-Controller design pattern concept. This was chosen because it allows the system to have the same data represented in different interfaces and be updated at the same time. We believed that this type of reactiveness was a great asset for this platform.

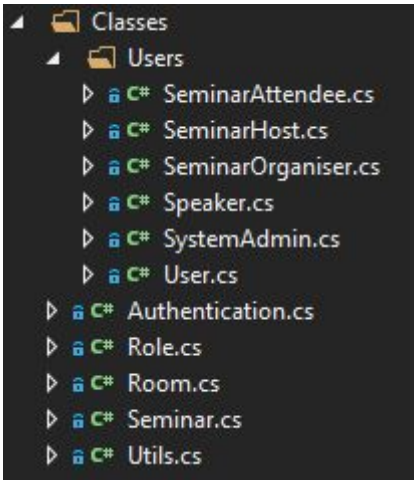
In order to integrate new features into this system, certain concepts and characteristics of the code base must be understood. These characteristics are broken down into four main categories:

1. Classes
2. Custom Controls
3. Forms
4. Resources



Classes

This section focuses on re-usable classes that represents a type of structure. These classes act as the Model for this system. The primary objective is to put light-weight class files in this section. For example, user types (Host, Organiser, etc). This section also contains very small modules that perform some basic manipulation such as retrieving all Speakers from a polymorphic list of users.

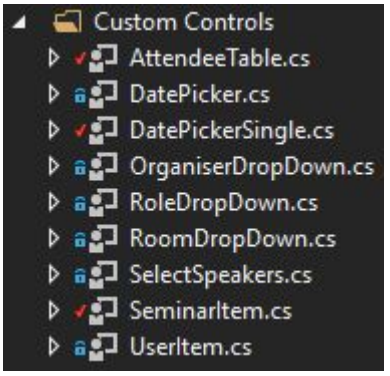


Due to the basic complexity of this project, there is only one layer of depth within the Classes category. This is the Users layer. This exists so the folder is more organised and manageable.

It is important to note that if the size of this project grows significantly, more subcategories can become existent. This will raise a potential issue of namespacing. It is crucial to be mindful of preserving a namespace and not polluting it with files from different domains. When creating a new file in Visual Studios it will by default, create a namespace based on the path to that file. For instance; creating a new file within Classes/Users will have the namespace of: *Seminar_Management_System.Classes.Users*

One module within this category named Utils, plays the role of providing system wide functionality. This is used for features such as creating a deep-copy of an object and retrieving controls from an interface.

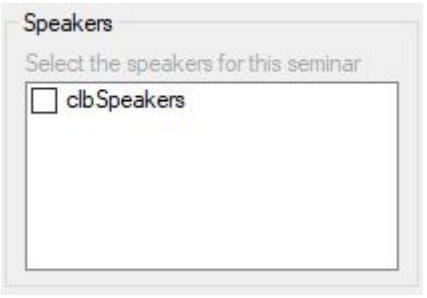
Custom Controls



In the ecosystem of this platform, a control is referred to as a piece in an interface. A common control is a button or text field. This project requires certain data to be visualised such as a Seminar, this should provide useful information at a glance. Another example is the need to ask a user to provide the system with information of a date and select a time whilst providing instant feedback to the user. This problem can be broken down into multiple components and created as a control which then can be reused throughout the project. This concept is the crux of Custom Controls.

Each Custom Control contains a visual design and a controller, these contribute to the View-Controller components of the Model-View-Controller design pattern. All controls should inherit from the UserController class.

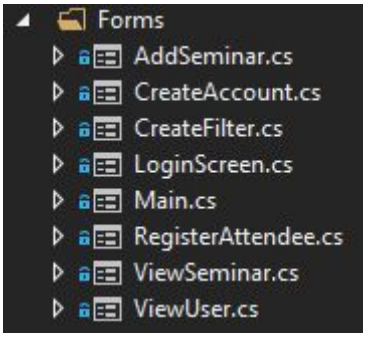
A Custom Control should be designed with the mindfulness of how that particular control will be used in multiple and different interface environments. This will assist the development process of the new control as it will expose potential flaws in code design. For instance, consider the SelectSpeaker control:



This control contains grey text which is used to indicate to the user some piece of information. To allow this control to be reused in different interfaces that don't necessarily adhere to its description, there should be code to enable this extendability and expose being able to modify its internal contents.

Forms

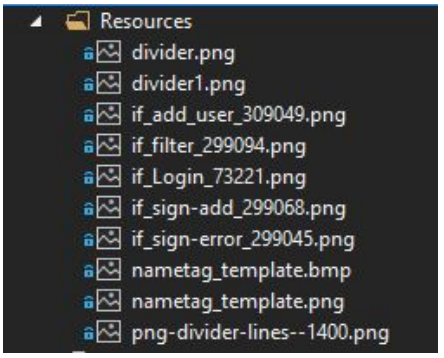
The Forms section is used for main interfaces. These interfaces are what comprise the entire front-end of the product. Each interface consists of common controls and Custom Controls. The primary objective of this section is to accurately reflect the state of the program in real time.



Interfaces interact with each other by reflecting data provided by the model. This ensures data changes occur everywhere when necessary. It is important for changes that occur internally to be influenced system-wide. Consider the following problem: An administrator logged in to the system that is viewing the details of a Seminar. Due to their higher access level, they can be editing the Seminar details in its entirety. Consider a scenario where the Admin doesn't close the Seminar page and then logs out of the system. Later, an Attendee uses the system and notices a page still open. If the system is unable to react to internal state changes of a different, less authorised User utilising the system then the high level editing features would still be available. This is why it is important to continue following the Model-View-Controller design patterns utilised in this section.

Resources

This section is used to centralise re-usable graphics. This is used by Visual Studios and C# to point to the same file whilst displaying it in multiple places. It is important to utilise this category when attempting to reduce the binary file of the final product. Consider compressing the images within this section.



General image types will be found in this section, such as PNG and JPG. Icons are not saved here because they are handled differently by Visual Studios and become embedded into the project.

Code Snippets

The following figures are snippets from the software to demonstrate the styling of the code base.

Room.cs

Seminar Management System | Seminar_Management_System.Classes.Room | Room(int id, string name, string location, int capacity)

```
1  namespace Seminar_Management_System.Classes
2  {
3      // A Room is a physical room that is used for having a Seminar in
4      public class Room
5      {
6          public int ID { get; set; } // Unique identifier
7          public string Name { get; set; } // Friendly Name
8          public string Location { get; set; } // Location eg: CB11.04.100
9          public int Capacity { get; set; } // How many people can fit in this room
10
11         public Room(int id, string name, string location, int capacity)
12         {
13             this.ID = id;
14             this.Name = name;
15             this.Location = location;
16             this.Capacity = capacity;
17         }
18     }
19 }
```

Utils.cs*

Seminar Management System | Seminar_Management_System.Classes.Utils

```
1  using Seminar_Management_System.Classes.Users;
2  using System;
3  using System.Collections.Generic;
4  using System.IO;
5  using System.Linq;
6  using System.Runtime.Serialization.Formatters.Binary;
7  using System.Windows.Forms;
8
9  namespace Seminar_Management_System.Classes
10 {
11     // This class is used for project wide re-usable functions
12     class Utils
13     {
14         // Returns all the Controls that match a certain type
15         // I.e.: All the text boxes from a form
16         static public IEnumerable<Control> GetControlsFromControl(Control control, Type type)
17         {
18             var controls = control.Controls.Cast<Control>();
19
20             return controls.SelectMany(ctrl => GetControlsFromControl(ctrl, type))
21                                        .Concat(controls)
22                                        .Where(c => c.GetType() == type);
23         }
24
25         // This is used to create a deep copy of an object
26         // It is used throughout the project to duplicate objects that are memory references
27         internal static class ObjectCloner
28         {
29             // Serialize and deserialize objects
30             public static object Clone(object obj)
31             {
32                 using (MemoryStream buffer = new MemoryStream())
33                 {
34                     BinaryFormatter formatter = new BinaryFormatter();
35                     formatter.Serialize(buffer, obj);
36                     buffer.Position = 0;
37                     object temp = formatter.Deserialize(buffer);
38                     return temp;
39                 }
40             }
41         }
```

InterfaceUnlocker.cs*

Seminar Management System | Seminar_Management_System.InterfaceUnlocker

```
11  namespace Seminar_Management_System
12  {
13      // This is used to decide what interface components should be exposed
14      // to the current logged in user
15      class InterfaceUnlocker
16      {
17          public void Watch()
18          {
19              // Subscribe to changes to the logged in user
20              DataInstance.LoggedInUserChanged += DataInstance_LoggedInUserChanged;
21          }
22
23          private void DataInstance_LoggedInUserChanged(object sender, EventArgs e)
24          {
25              // Update the interface based on the new privilege of the logged in user
26              int priv = DataInstance.LoggedInUser.Role.Privilege;
27              this.UserList(priv);
28              this.AddSeminar(priv);
29              this.EditSeminar(priv);
30
31          }
32          private void EditSeminar(int priv)
33          {
34              // We force every ViewSeminar to restore its state
35              // The user then must re-enable Edit mode
36              // This will cause a new validation check to execute that is located in:
37                  // ViewSeminar.cs private void btnEdit_Click(object sender, EventArgs e)
38              // Based on this validation, the user will be able to edit all fields or only the attendee field
39              foreach (ViewSeminar sem in DataInstance.seminarInterfaceWindows)
40                  sem.RestoreState();
41          }
42
43          private void AddSeminar(int priv)
44          {
45              // Organisers and above can create seminars
46
47              DataInstance.mainInstance.btnAddSeminar.Visible =
48                  priv >= Authentication.GetPrivilegeFromRoleName(Role.Names.Organiser);
49          }
50
51          private void UserList(int priv)
52          {
53              // Only Admins can see the UserList interface
54
55              if (priv >= Authentication.GetPrivilegeFromRoleName(Role.Names.Admin))
56              {
57                  // Add user list page if its not there
58                  if (!DataInstance.mainInstance.tabControl.TabPages.Contains(DataInstance.mainInstance.userTab))
59                      DataInstance.mainInstance.tabControl.TabPages.Add(DataInstance.mainInstance.userTab);
60              }
```

```
66  public static class PortableFilter
67  {
68      // Each section will have a flag followed by a value
69
70      // Search by room
71      public static bool ByRoom { get; set; } // Enable or Disable searching by Room
72      public static Room Room { get; set; } // Which Room to search for
73
74      public static bool ByOrganiser { get; set; }
75      public static SeminarOrganiser Organiser { get; set; }
76
77      public static bool BySpeaker { get; set; }
78      public static List<Speaker> Speakers { get; set; }
79
80      public static bool ByDate { get; set; }
81      public static DateTime StartDate { get; set; }
82      public static DateTime EndDate { get; set; }
83
84
85      // Returns a list of Seminars that matches the search conditions defined by the properties above
86      public static List<Seminar> Execute()
87      {
88          var query = from s in DataInstance.seminars
89                      where (ByRoom == false || s.Room == Room)
90                      where (ByOrganiser == false || s.Organiser == Organiser)
91                      where (BySpeaker == false || s.Speakers.Any(iter => Speakers.Any(speaker => speaker.Name == iter.Name)))
92                      where (ByDate == false || (s.StartDate >= StartDate && s.StartDate <= EndDate))
93                      select s;
94          return query.ToList<Seminar>();
95      }
96  }
97  }
98
```

UserItem.cs

Seminar Management System — Seminar_Management_System.Custom_Controls.UserItem

```
14  namespace Seminar_Management_System.Custom_Controls
15  {
16      // This is used to graphically represent a User
17      public partial class UserItem : UserControl
18      {
19          public UserItem()
20          {
21              InitializeComponent();
22          }
23
24          public User UserReference;
25          public bool isUsingAlternativeColor = false; // Used to alternate shades of the same color
26          public Color BackgroundColor { get { return this.BackColor; } set { this.BackColor = value; } }
27          public void Populate(ref User user)
28          {
29              // Connect to the User reference
30              UserReference = user;
31              lblName.Text = user.Name;
32              lblRole.Text = user.Role.Name;
33              CustomBackgroundColor(true);
34
35          }
36          public void AlternativeColor()
37          {
38              CustomBackgroundColor(false);
39              this.isUsingAlternativeColor = true;
40          }
41          private void CustomBackgroundColor(bool defaultShade = true)
42          {
43              // defaultShade is used to determine if the color should be alternated
44              // Define what color is reflected by which role
45              switch (UserReference.Role.Name)
46              {
47                  case Role.Names.Admin:
48                      this.BackColor = defaultShade ? Color.Orange : Color.DarkOrange;
49                      break;
50                  case Role.Names.Host:
51                      this.BackColor = defaultShade ? Color.PaleGreen : Color.LightGreen;
52                      break;
53                  case Role.Names.Organiser:
54                      this.BackColor = defaultShade ? Color.MediumPurple : Color.BlueViolet;
55                      break;
56                  default:
57                      this.BackColor = defaultShade ? Color.LightCoral : Color.Salmon;
58                      break;
59              }
60          }
61          private void UserItem_Load(object sender, EventArgs e)
62          {
63              this.Resize();
64          }
```

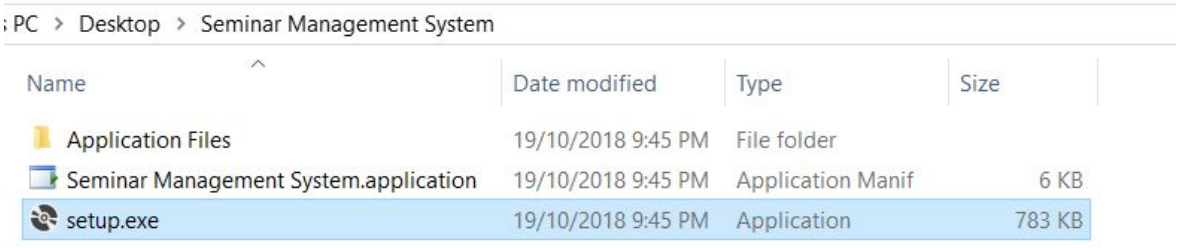
Purpose

The purpose of having a user manual is so that if the end user has troubles either installing the program or using it, this is a backup that can inform the user how to complete the function they were struggling with. The application should be intuitive enough to not need a user manual, but this is here in case there are any problems. The user manual should be simple and easy to understand, and inform the user about all functionality of the program.

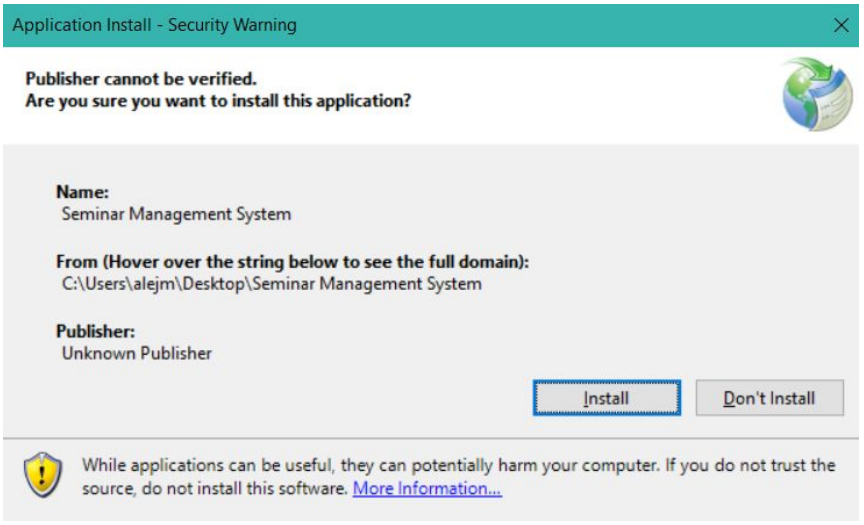
Installation

To install the seminar management system follow these steps:

1. Download from website



2. Run "Setup.exe"
3. Click install

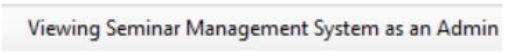


4. The software should automatically start running

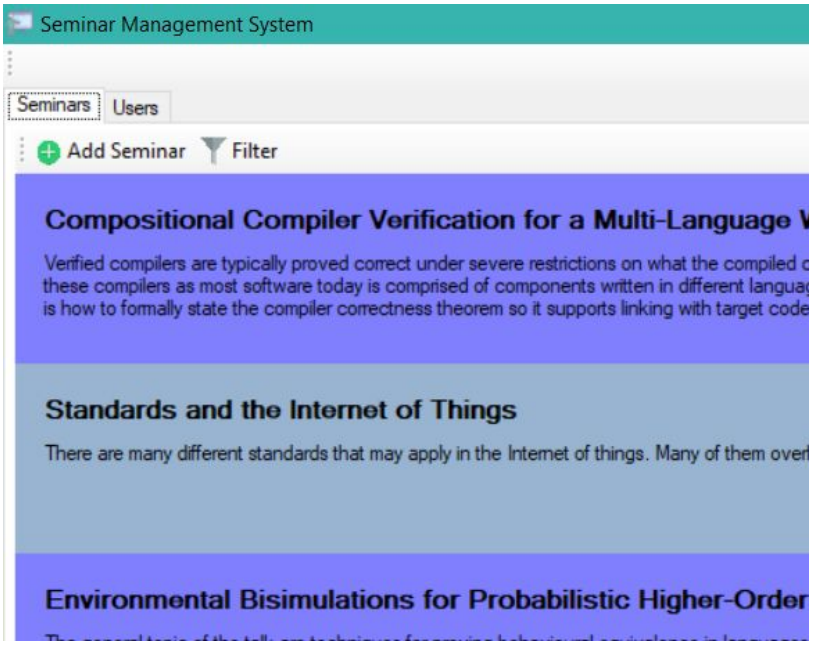
To run the software after installation simply run the "Seminar Management System.application" file

Add a seminar

In order to add a seminar you must be logged in as either an organiser or an admin. You can see which user you are logged in as at the bottom right hand corner of the screen.



To create a new seminar just click the green plus icon in the top left corner of the screen.

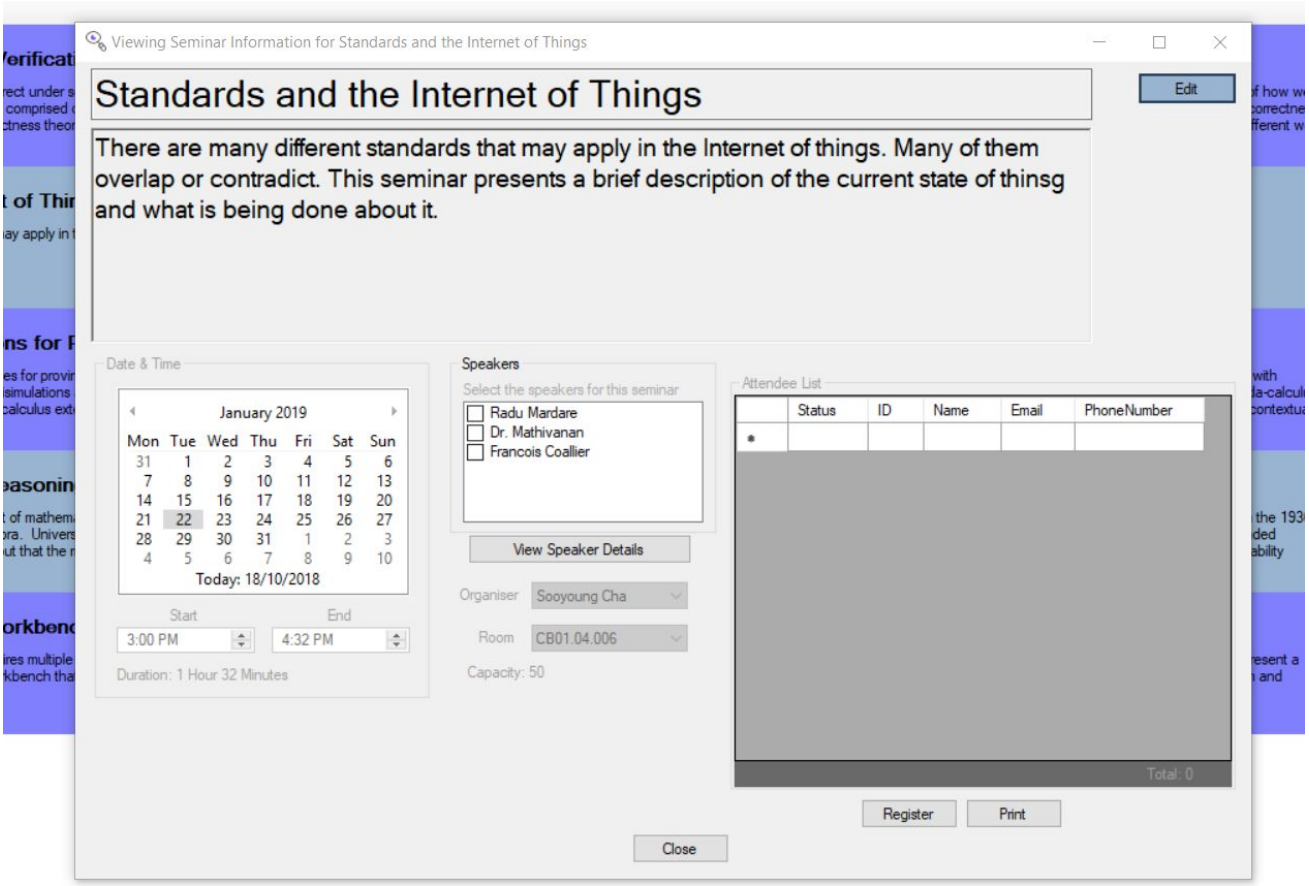


Filter by Seminar

In the top left corner there is also an option to filter the seminars by their room, organiser, speaker or date.

View Seminar

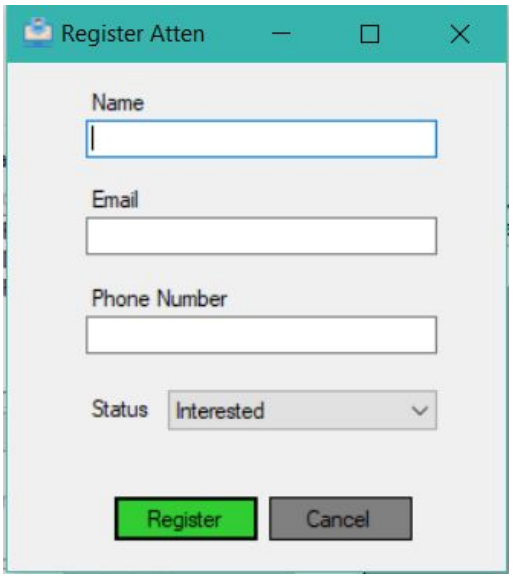
To view a seminar click on the seminar that you wish to view and a view seminar screen should appear before you.



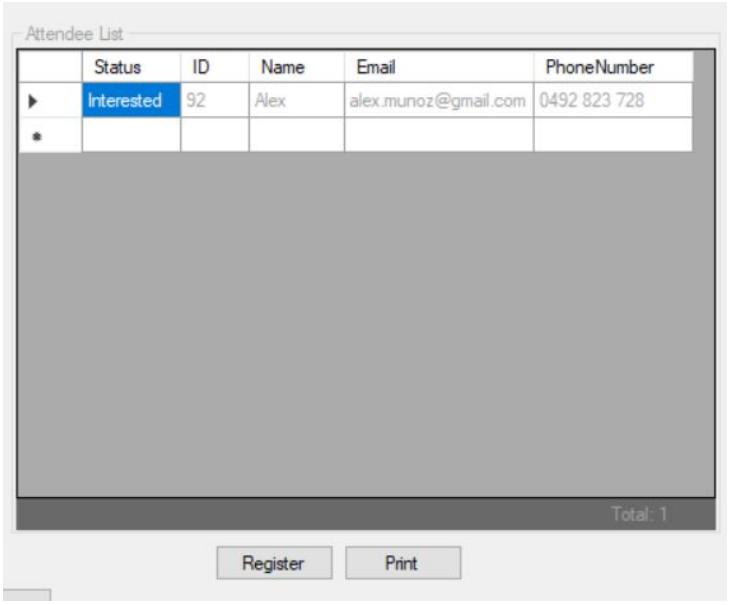
On this screen you can see the Title up the top, the description of the seminar (below the title) the date, time and duration of the seminar, the speakers speaking at this seminar, the location and capacity of location of the seminar, the seminar organiser and finally the attendees that are attending the seminar.

Register Attendee

To register an attendee to the seminar that you are currently viewing, click the register button that is just below the attendee list:



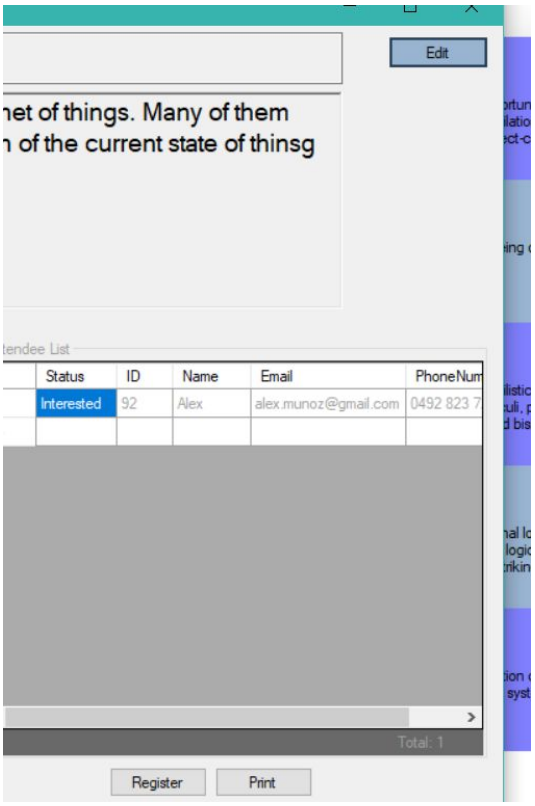
A small pop out screen shown above should appear, in this screen you can record your name, email and phone number to receive updates on the seminar and reminders. You can also select whether you are interested in attending the seminar or going to the seminar. A small message screen should appear to confirm that you have registered your interest in the seminar.



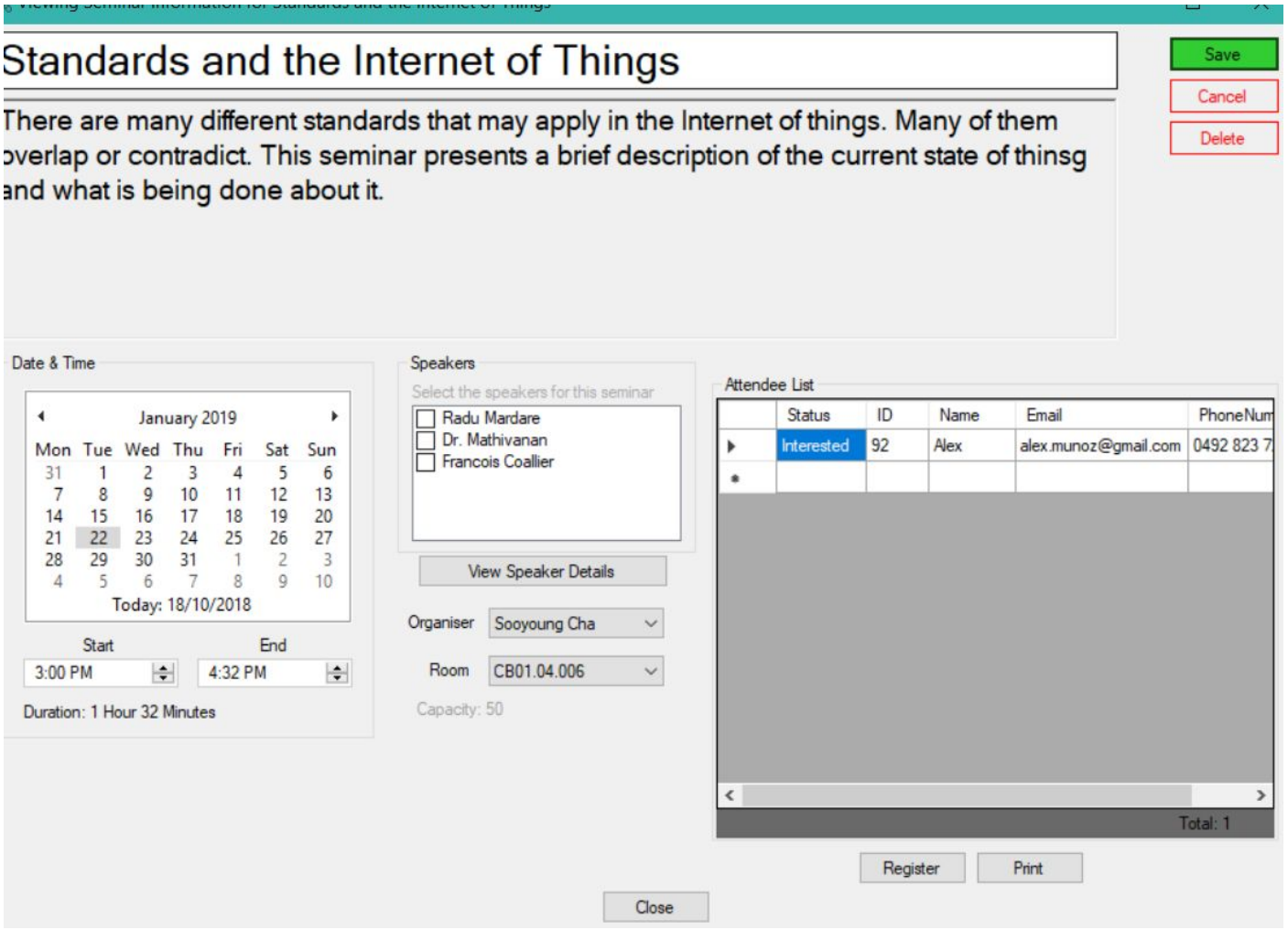
You should also notice that your name, email and phone number are now in the attendee list. Note that this information is now publicly available to other attendees.

Edit Seminar

If you are logged in as either an Admin or Seminar Organiser you can edit the seminar you are viewing by clicking the edit button in the top right hand corner.



You can see that you are in edit mode because the buttons will change and all the fields will become editable.



Once you have finished editing any fields, you can click the Save button to save the seminar in its current state, or if you made a mistake you can press cancel to return the seminar to its previous state.

Deleting Attendee

To delete an attendee that has been registered to the seminar, you can press the edit button then select the row of the attendee, then press the delete button on your keyboard.

If you would want to delete the seminar you can also do this by pressing the Delete button in the top right corner.

Print Attendee Name Tags

To print out a list of the attendees name tags, you can click the Print button. This creates a PDF which you can save and then print later.

View Users

If you are logged in as an administrator you also have access to all the users in the system. Admins are orange in colour, hosts are green, organisers are purple and speakers are pink.

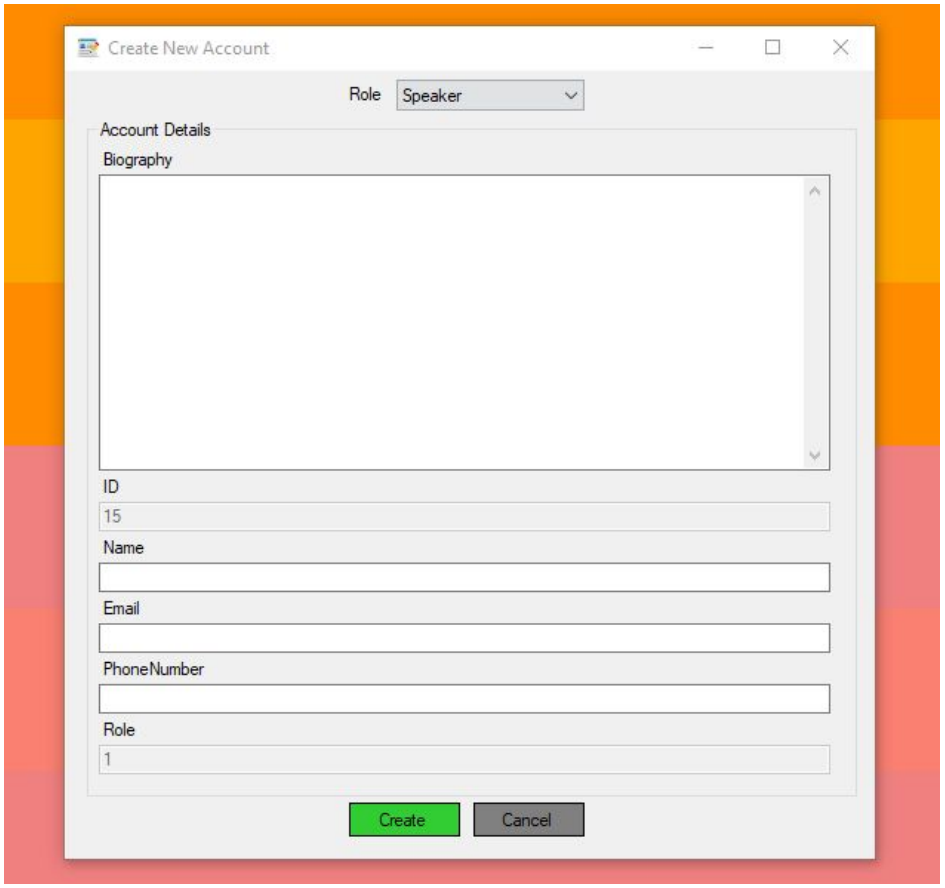
To view any of the users details, you can click on the user. This will bring up a screen that shows their details.

Edit a users details

Similar to editing a seminar, you can edit a user by clicking on the edit button. This makes the fields editable and then you can click save to save the state of that user or cancel if you wish it to return to its previous state. If you wish to delete the user you can click the delete button.

ADD NEW USER

To add a user, simply click the green plus icon at the top left hand corner and a Create New Account screen will appear. See the figure below.



Here you can select what type of user you wish to create, then you can fill in the fields that are required. After creating a user a confirmation box should appear to confirm that your user was created. Then you should be able to see the user in the user list.

TEST STRATEGY

INTRODUCTION

This test strategy is an outline that describes the testing approach of the software development cycle for our seminar management system. We use it to inform our project manager and test team about the key issues of the testing process. Overall, this document is designed based on the project plan and requirement specification, making sure our product is high quality, on-time deliverable and to meet all the customer requirements. There are several steps of testing methods: unit testing, integration testing, system testing and acceptance testing. In our plan, we test our system following these 4 testing levels step by step, and achieve a high quality seminar management system which fully meets all the requirements.

PURPOSE OF TESTING

To provide a high quality product to the customer, it is critical to implement testing when develop an application. By implementing the testing we can point out the defects and errors that were made during the development phases. It prevents our system crashing because of the undetected bugs or invalid actions. A well designed testing strategy and implementation also provides confidence and reliability to our customer. As the customer, they ask for a product which should be low cost, reliable and effective performance. If we fail to properly test our software at this stage, it could result in critical failures later on in the development process. It is our responsibility to find out the potential defects, unsatisfactory user experience, as well as bad performance. We need to have our system repeatedly tested following our test strategy to ensure that the software that is delivered to the client is up to standard.

TEST OBJECTIVES

The main objective of this test strategy is to verify whether the functional and the non-functional requirements of our system have been implemented properly. We also aim to test whether the system's user experience is satisfactory and that the system is accessible to all users.

The final ideal test result would be :

1. The functionality meets the client's requirements
2. The authorization meets the client's requirements
3. Potential defects are discovered and resolved with due diligence

4. The user experience is satisfactory to the client

5. The non-functional requirements meets the client's requirement

6. The database has been seamlessly integrated into the system properly with no issues

Assumptions for Testing

Technical(key) Assumptions:

1. All the required functions have already been fully implemented before the test.
2. The tests are conducted on Windows 10 operating system.
3. Tester will follow the test cases to test the system.
4. System has already been successfully connected to the system.

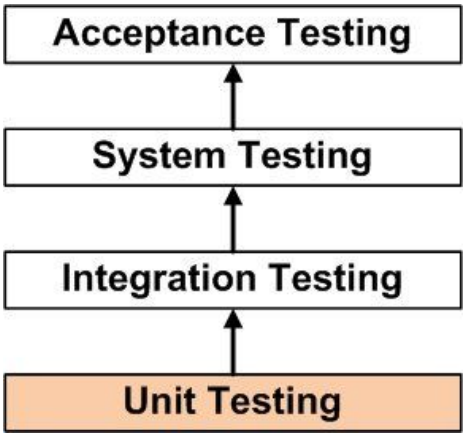
General Assumptions:

1. Tester is familiar with the system controls
2. Tester is fluent in both reading and writing in English
3. There are seminars that have been pre-loaded into the system.
4. There are organisers already entered into the system.
5. All roles included in the system are usable.

Test Scope

1. Both the valid and invalid data input will be tested while testing. This ensures each function is working logically and also reasonably. Help avoiding application crashed by invalid input.
2. The test for user interface design and user experience are also included. This is testing while tester is doing the functionality test and will be carried out by tester.
3. The database will also be tested by checking the data inside.
4. The performance of the system is also tested.
5. The unpredictable actions are also considered and tested. This ensures the application won't get crashed or lose data because of accidents such as shutting down computer.

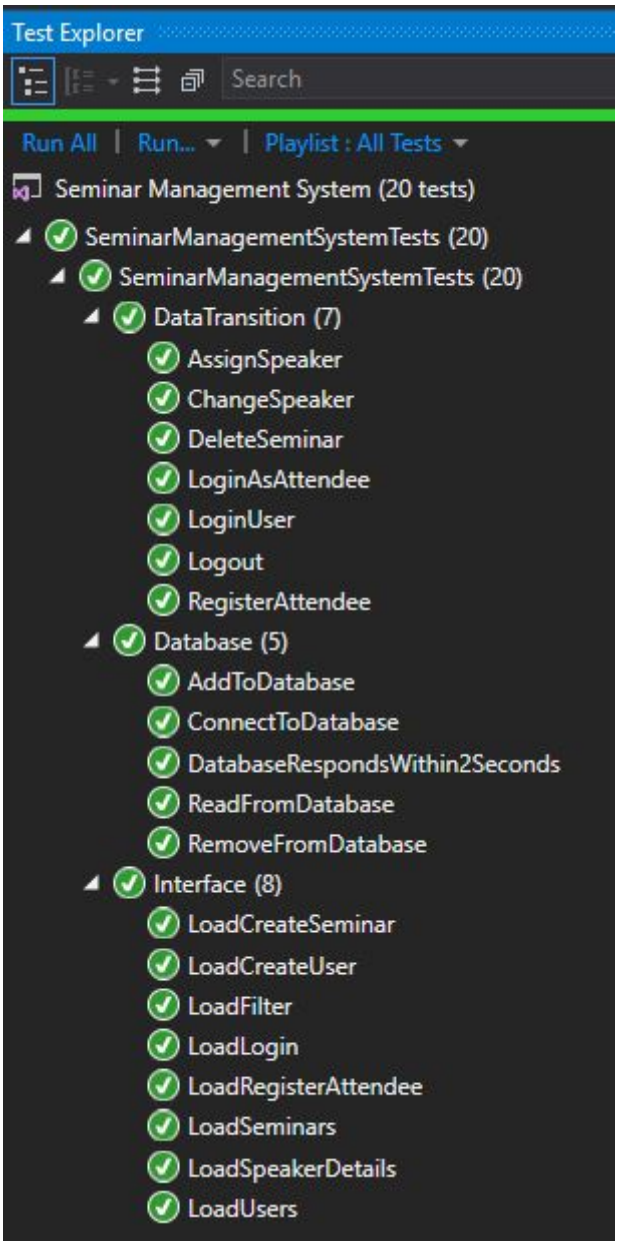
UNIT TESTING

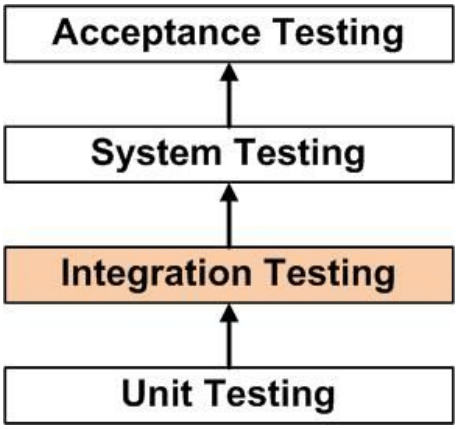


Unit testing is a level of software testing where individual units or components of a software are tested. The reason why we implement unit testing is to validate that each unit of the software worked as we designed. A unit is the smallest part in software, which usually has few inputs and a single output. It can be a function, method, etc. By implementing the unit testing, the confidence in maintaining code will increased. It costs less to fix defects because we can detect any small defects in a very low level, rather than detect at the last minutes when we have nearly completed the software. Besides, writing the unit testing helps us to increase our efficiency of development. Rather than set up a break point, we provide some input manually and consider different situations each time. While writing unit testing takes up time initially, it will save us time later on in development. We do not need to start and stop the program to check for defects and makes debugging easily. Only the latest changes need to be debugged when the test fails.

We have utilised unit testing in our software in order to detect breaking code changes. The screenshot below shows the sample unit tests for each unit. Roughly we divide our unit tests into three types. DataTransition manages the data communication among each unit. Database manages the unit related to database. Interface manages the unit which is connected with the interface. By assigning unit test to each small unit (in our case is mostly method), we can see what is going on each unit and lock the defect in a very small scope. For example, all the unit tests will run when we run the application. Depends on the use of this unit, we set different input for this unit and check if the output is what we desire. If the output correct, the related unit test will return true and pass. Once there is a failed unit test, such as AddToDatabase. We can immediately know there is something wrong in AddToDatabase method. As a result, instead of looking through the whole system to see where is going wrong we can just simply go to AddToDatabase and try to fix it. If all the tests run perfectly but crashed some of them after we making some new changes, we can also understand that the reason why those tests fail must be something

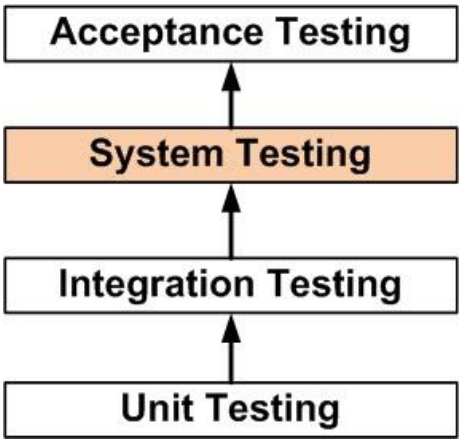
wrong in the new change so that we can avoid detecting this defect after we have already made many new changes to the code, which is much more inefficient.





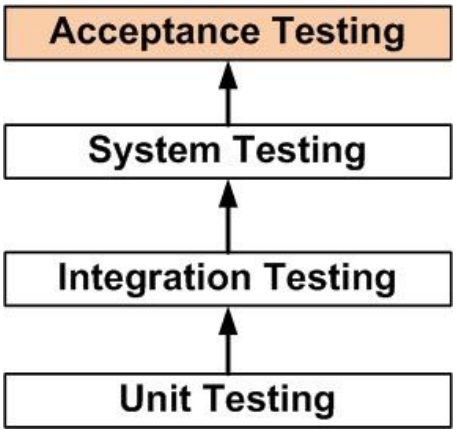
Though we have already implement several unit tests to test each of units, we are not sure whether there is any new defects if they combined and tested together. We need to implement integration testing. Integration testing is a level of software testing where individual units are combined and tested as a group. We implement the integration test is to expose faults in the interaction between integrated units. Instead of writing test code like unit testing, all of our team developer act as testers to perform integration testing. We believe that test by developer would cover more potential bugs in this level.

There several approach to implement the integration testing which is Big Bang, Top Down, Bottom Up and Sandwich. Comparing each of these approaches, we first decided not to follow Big Bang to implement integration testing because it needs all or most of the units are combined together and tested at one go. We all agreed that it is high risk to test integration unitil the entire system comes out because of the time and cost limitation. It is hard for us to fix the bugs in this level when the entire system nearly comes out. Due to our system is developed from the Main form which displays a list of seminars, we decided to use Top Down approach rather than others. We tested the interaction between the Main form and other child forms such as AddSeminar to see whether the connection works properly. In the scope integration testing, we tested the things such as value passing, windows moving, etc. For example, tester run the system, open the Main window and click the view button in order to transfer to the SeminarDetails window. The expected output would be SeminarDetails window shows up and displays the correct interface and data.



System testing is the third level of software testing. The purpose of this test is to run the system and test it whether it meets specified requirements. Tester will run the application, following the test case document and test the system step by step. Without considering things such as UI design, only the functional requirements will be tested in this level. We require tester to follow the function test cases to test the requirements. The expected output of this system testing is to pass all the test cases, which means our application can satisfy the functional requirements.

Acceptance Testing



Acceptance testing is the highest level of software testing. The purpose of acceptance testing is to judge whether the application is acceptable for delivering to customers. In this level, the system should be tested not only the functional requirements but also the things such as business requirements, user experience, ui design, usability, etc. Our application must pass this test before delivering to customers. The tester of acceptance testing would be the non-developer in our team. To pass the acceptance testing, the application must pass the acceptance test table and pass the tester feeling test. Tester feeling test is about telling the user experience by tester. An acceptable application should let user feel comfortable to use it. The acceptance test table is shown on the link at the next session. Only these two test passed, our application can be know as acceptable enough to deliver to customers.

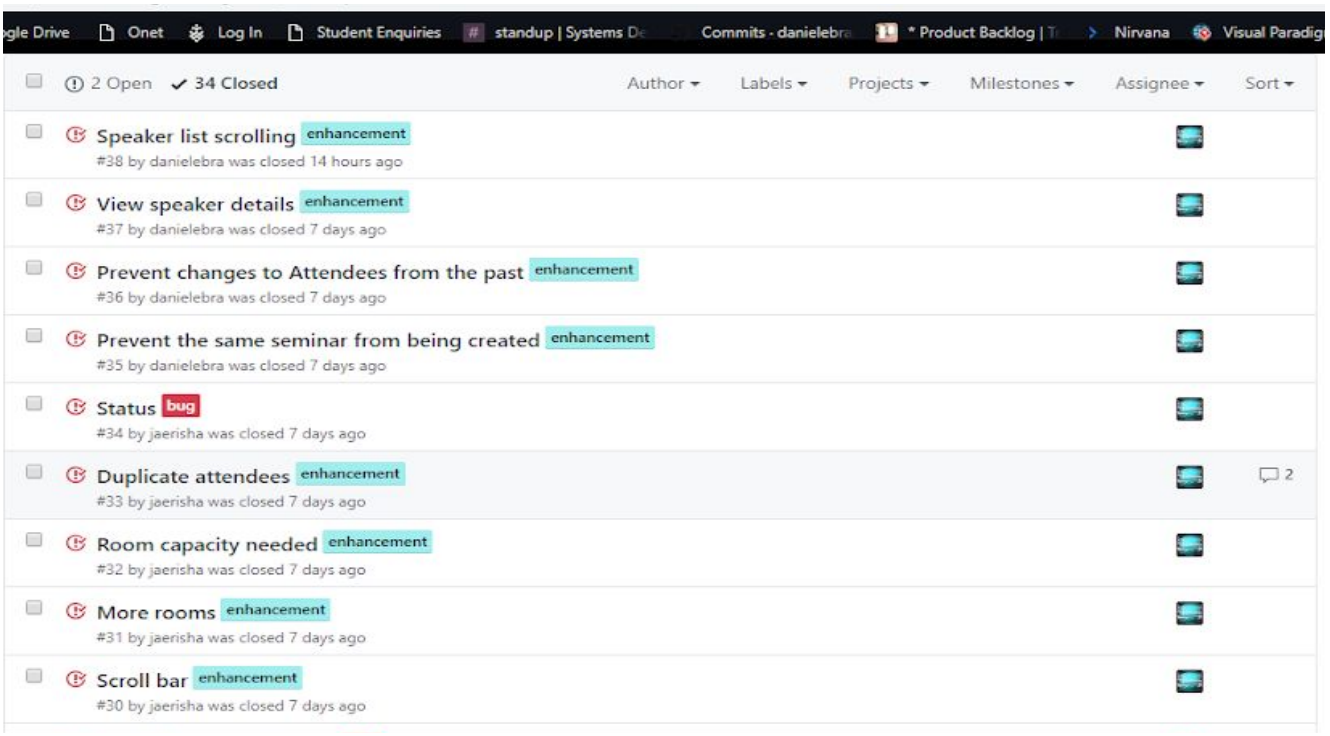
Acceptance Test Table and Function Test Cases

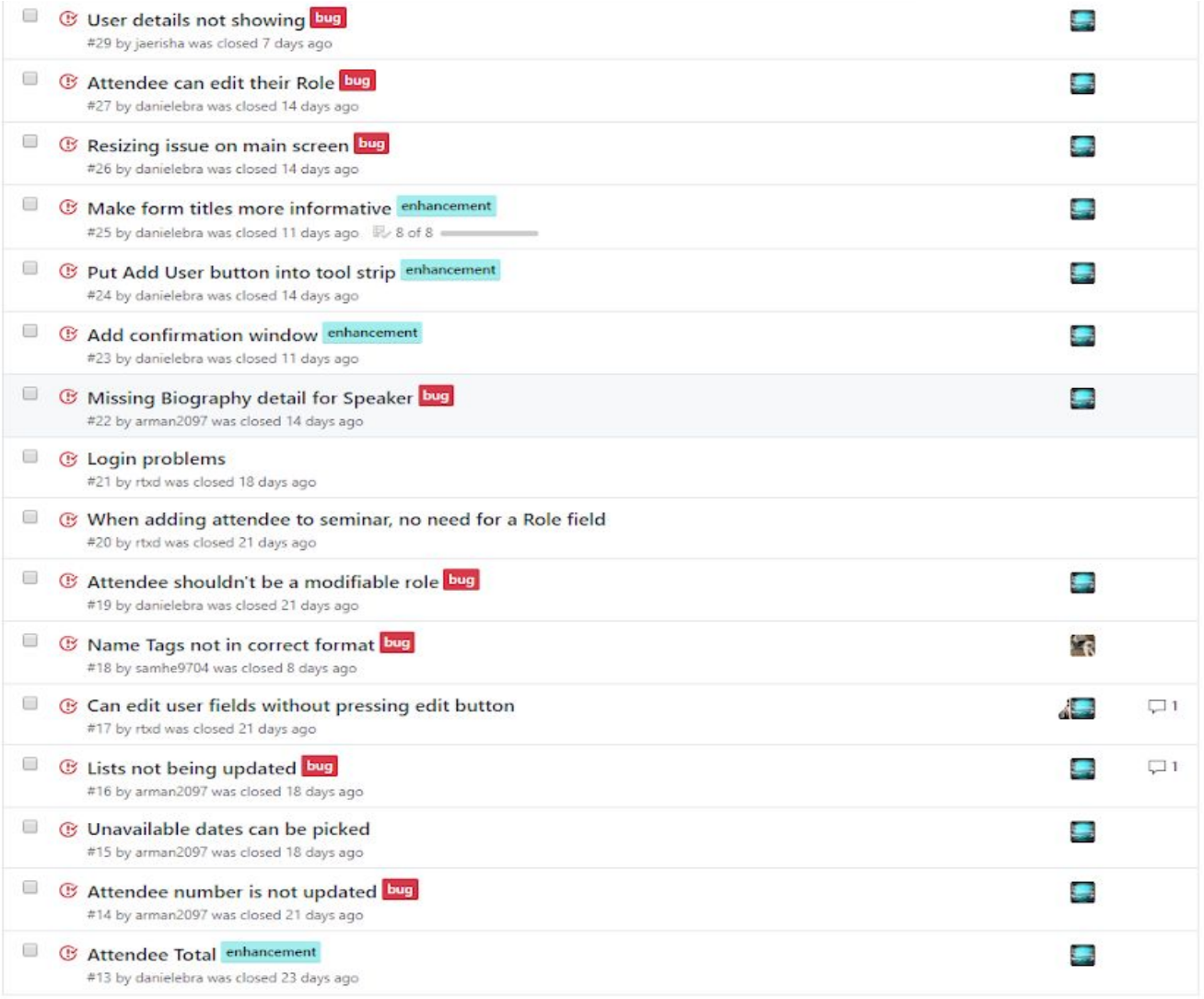
Throughout the process of acceptance testing we documented our results in an acceptance test table. This can be accessed through here.

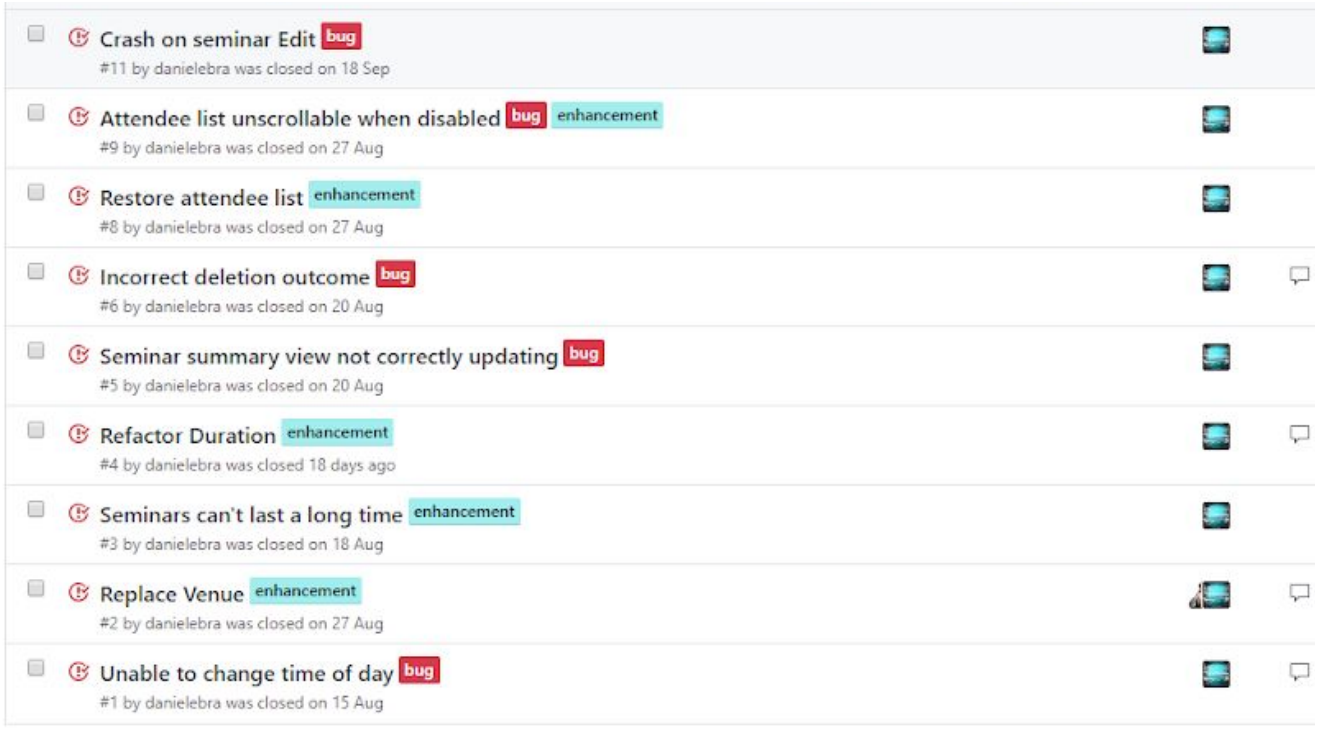
Defect Tracking Process

In order to develop a complete and functional system, we have to make sure it is free of errors and has met all the client's requirements. To achieve this, we frequently tested the system in order to find bugs and defects in the system. Any error found while testing goes through the defect life cycle (explained here), starting from status new all the way to closed. When an error is found it a ticket is opened in Github issues and later in the group chat. Following the opening of the ticket it would move on to the assigned phase when we assigned a member to solve it. Following this it would exist in the open phase. The assigned member would then solve the issue and begin the phase of retesting. Once the tester was satisfied with the results, the ticket would be closed and the defect would finally reach the closed phase of the defect life cycle.

Through this process we are ensuring that we have a functional system that has met all the requirements of the clients. Attached below are the screenshots of some of the issues that has been resolved and closed.




User details not showing bug
#29 by jaerisha was closed 7 days ago
Attendee can edit their Role bug
#27 by danielebra was closed 14 days ago
Resizing issue on main screen bug
#26 by danielebra was closed 14 days ago
Make form titles more informative enhancement
#25 by danielebra was closed 11 days ago 8 of 8
Put Add User button into tool strip enhancement
#24 by danielebra was closed 14 days ago
Add confirmation window enhancement
#23 by danielebra was closed 11 days ago
Missing Biography detail for Speaker bug
#22 by arman2097 was closed 14 days ago
Login problems
#21 by rtxd was closed 18 days ago
When adding attendee to seminar, no need for a Role field
#20 by rtxd was closed 21 days ago
Attendee shouldn't be a modifiable role bug
#19 by danielebra was closed 21 days ago
Name Tags not in correct format bug
#18 by samhe9704 was closed 8 days ago
Can edit user fields without pressing edit button
#17 by rtxd was closed 21 days ago
1
Lists not being updated bug
#16 by arman2097 was closed 18 days ago
1
Unavailable dates can be picked
#15 by arman2097 was closed 18 days ago
Attendee number is not updated bug
#14 by arman2097 was closed 21 days ago
Attendee Total enhancement
#13 by danielebra was closed 23 days ago


Crash on seminar Edit bug
#11 by danielebra was closed on 18 Sep
Attendee list unscrollable when disabled bug enhancement
#9 by danielebra was closed on 27 Aug
Restore attendee list enhancement
#8 by danielebra was closed on 27 Aug
Incorrect deletion outcome bug
#6 by danielebra was closed on 20 Aug
Seminar summary view not correctly updating bug
#5 by danielebra was closed on 20 Aug
Refactor Duration enhancement
#4 by danielebra was closed 18 days ago
Seminars can't last a long time enhancement
#3 by danielebra was closed on 18 Aug
Replace Venue enhancement
#2 by danielebra was closed on 27 Aug
Unable to change time of day bug
#1 by danielebra was closed on 15 Aug

While the use of Github issues has been good for tracking defects in the system, as the repository is private, we are not able to share a direct link to the issues. Thus, we decided to also document the bugs in a defect log. During the creation of the defect log we found that the visualisation of open and closed issues on the same screen made it easier to see how many issues we have versus how many have been resolved. Below is a figure of the current defect log.

Defect ID	Title	Description	Category	Severity	Created by	Created date	Assigned to	Resolved date	Resolved by	Status
D001	Unable to change time of day	Date and Time picker can't be modified between AM and PM when typing input via the key	Bug	H	Daniel E.	15/08/2018	Daniel E.	15/8	Daniel E.	Closed
D002	Replace venue	Based on the information provided by the subject coordinator. All references to Venue will	Enhancement	L	Daniel E.	16/08/2018	Daniel E.	27/8	Daniel E.	Closed
D003	Seminars can't last a long time	Based on customer feedback, a seminar will last less than a day. Usually for one hour. Bei	Enhancement	L	Daniel E.	18/08/2018	Daniel E.	18/8	Daniel E.	Closed
D004	Refactor duration	The DatePickerSingle is now being used in multiple places. It would benefit from duration I	Enhancement	L	Daniel E.	19/08/2018	Daniel E.	30/9	Daniel E.	Closed
D005	Seminar summary view not correctly updating	Old seminar items seem to be remaining on screen even when they are modified.	Bug	H	Daniel E.	20/08/2018	Daniel E.	20/8	Daniel E.	Closed
D006	Incorrect deletion outcome	When editing a seminar then adding new ones, it appears that everything acts as intended However, when deleting the same seminar, it appears that another seminar is deleted inste	Bug	H	Daniel E.	20/08/2018	Daniel E.	20/8	Daniel E.	Closed
D007	Sample data	Sample data is located inside yellow book case study, update database with it	Enhancement	M	Alex M.	23/08/2018	Alex M.			Open
D008	Restore attendee list	If someone enables editing mode and changes a field of a Attendee then decides to press	Enhancement	L	Daniel E.	26/08/2018	Daniel E.	27/8	Daniel E.	Closed
D009	Attendee list is unscrollable when disabled	When AttendeeTable is disabled, it can't be scrolled. When it is enabled, it can be scrolled.	Bug/Enhancement	M	Daniel E.	26/08/2018	Daniel E.	27/8	Daniel E.	Closed
D010	Connect to AWS database	Utilise an SqlConnection to retrieve data from a Microsoft SQL Server that is hosted by An	Pull Request	H	Daniel E.	01/09/2018	Daniel E.	01/09/2018	Daniel E.	Closed
D011	Crash on seminar edit	The program crashes when pressing the Edit Seminar button.	Bug	H	Daniel E.	18/09/2018	Daniel E.	18/9	Daniel E.	Closed
D012	Add Comments to project files	Comments were requested to be added to each file in the project.	Pull Request	H	Daniel E.	23/09/2018	Daniel E.	26/09/2018	Daniel E.	Closed
D013	Attendee total	Attendee Table should display the total number of entries.	Enhancement	L	Daniel E.	26/09/2018	Daniel E.	26/9	Daniel E.	Closed
D014	Attendee number is not updated	Total number of atttendees is not updated when a row is deleted.	Bug	H	Daniel E.	27/09/2018	Daniel E.	27/9	Daniel E.	Closed
D015	Unavailable dates can be picked	When creating a seminar dates that should be unavailable can be picked	Bug	H	Kazi A.	27/09/2018	Daniel E.	1/10	Daniel E.	Closed
D016	Lists not being updated	When a speaker is regisgtered it doesn't show up in the list while choosing a speaker for a	Bug	H	Kazi A.	27/09/2018	Daniel E.	1/10	Daniel E.	Closed
D017	Can edit user fields without pressing edit button	Users shouldn't be able to edit the fields of another user if they haven't pressed the edit bu	Bug	H	Alex M.	27/09/2018	Daniel E., Alex M.	27/9	Daniel E.	Closed
D018	Name tags are not in the correct format	when printing the tags they are not in the correct position	Bug	H	Sam H.	5/10/2018	Sam H.	11/10	Sam H.	Closed
D019	Attendee shouldn't be a modifiable role	New users shouldn't have the option to have their account created as an Attendee.	Bug	H	Daniel E.	27/09/2018	Daniel E.	27/9	Daniel E.	Closed
D020	When adding attendee to seminar, no need for a Role field	Hiding the role field	Enhancement	L	Alex M.	27/09/2018	Daniel E.	27/9	Daniel E.	Closed
D021	Login problems	When I log into the system I need some visual feedback that i've been logged in successfu Maybe change the button to a log out button instead, and when you are logged in maybe a	Enhancement	L	Alex M.	27/09/2018	Daniel E.	27/9	Daniel E.	Closed
D022	Missing biography details for speaker	When viewing the details of a Speaker, the biography field is not displayed	Bug	H	Kazi A.	4/10/2018	Daniel E.	4/10	Daniel E.	Closed
D023	Add confirmation window	Create a confirmation window to pop up when the user clicks ok on the View Seminar pag	Enhancement	L	Daniel E.	4/10/2018	Daniel E.	8/10	Daniel E.	Closed
D024	Put "Add User" button into tool strip	The Add User button that appears on the View Users tab page needs to be put into its owr	Enhancement	L	Daniel E.	4/10/2018	Daniel E.	4/10	Daniel E.	Closed
D025	Make form titles more informative	The form titles are currently mostly set as default titles. They can be made more informativ	Enhancement	L	Daniel E.	4/10/2018	Daniel E.	8/10	Daniel E.	Closed
D026	Resizing issue on main screen	When resizing the screen and the current selected tab is for example on, Seminars. The n	Bug	M	Daniel E.	5/10/2018	Daniel E.	5/10	Daniel E.	Closed
D027	Attendee can edit their Role	When editing the Attendee Table, the user is exposed to the Role column. Editing this colu	Bug	H	Daniel E.	5/10/2018	Daniel E.	5/10	Daniel E.	Closed
D028	Attendee Status appears before ID	The data table for the AttendeeStatus is placing Status as the first column. This should be moved, perhaps after the Name column.	Enhancement	L	Daniel E.	5/10/2018	Daniel E.			Open
D029	User details not showing	After a user is added by phone, email, etc, it does not show the user details when clicked (	Bug	H	Jelisha D.	11/10/2018	Daniel E.	11/10/2018	Daniel E.	Closed
D030	Scroll bar	There should be a scroll bar in the speaker's biography	Enhancement	L	Jelisha D.	11/10/2018	Daniel E.	11/10/2018	Daniel E.	Closed
D031	More rooms	There should be more places to choose from than Building 10 and 11	Enhancement	L	Jelisha D.	11/10/2018	Daniel E.	11/10/2018	Daniel E.	Closed
D032	Room capacity needed	There is no room capacity in our software	Bug	H	Jelisha D.	11/10/2018	Daniel E.	11/10/2018	Daniel E.	Closed
D033	Duplicate attendees	Need to get rid of duplicates	Enhancement	L	Jelisha D.	11/10/2018	Daniel E.	11/10/2018	Daniel E.	Closed
D034	Status	Changing the status options should not be allowed	Bug	H	Jelisha D.	11/10/2018	Daniel E.	11/10/2018	Daniel E.	Closed
D035	Prevent the same seminar from being created	Prevention for creating the same seminar must be put in place.	Enhancement	L	Daniel E.	11/10/2018	Daniel E.	11/10/2018	Daniel E.	Closed
D036	Prevent changes to Attendees from the past	A check should be added that will only allow editing to the Attendee list when it is present (	Enhancement	L	Daniel E.	11/10/2018	Daniel E.	11/10/2018	Daniel E.	Closed
D037	View speaker details	Speaker details such as biography need to be exposed when viewing a seminar so that ar	Enhancement	L	Daniel E.	11/10/2018	Daniel E.	11/10/2018	Daniel E.	Closed
D038	Speaker list scrolling	The SelectSpeaker custom control can't be scrolled when editing is disabled.	Enhancement	L	Daniel E.	18/10/2018	Daniel E.	18/10/2018	Daniel E.	Closed
D039	Add name tag printing		Pull Request	H	Daniel E.	18/10/2018	Daniel E.	18/10/2018	Daniel E.	Closed
D040	Integrating database		Pull Request	H	Alex M.	18/10/2018	Alex M.	18/10/2018	Daniel E., Alex M.	Closed
	Total Amount of Issues:	40								
	Unresolved	2								
	Resolved	38								
	% Resolved	95								

The defect log reflects the Github issues page's issues in the order that they were created. The different colours show what status the bugs currently are. The green represents resolved tickets, while red represents unresolved tickets. The purple shows pull requests that have been done. Also shown on the defect log are all the details regarding the tickets, including the description, assigned member and dates of when the ticket was opened or closed. Having the information laid out on one page makes it simple to see the details of all the defects. In contrast, navigating through Github issues can be frustrating for members even though there is more detail shown there that is not included in the defect log.

Below the defect log there are also some statistics to show the percentage of resolved tickets versus unresolved tickets. While the visual representation is good for members to see approximately how many tickets have been resolved it is useful to have exact figures when it comes to documentation of the development process.

Test Case ID	Process	Expected Outcome
Test Case 1: Login Function		
TC01-01	Login with correct account and password	User login into the system
TC01-02	Login with incorrect password	Login fails
TC01-03	Login with incorrect account	Login fails
Test Case 2: Display summary of Seminar		
TC02-01	View the Seminar list	All Seminars appear
TC02-02	Search Seminars by Venue	Only Seminars for that specific Venue appear
TC02-03	Search Seminars by Date	Only Seminars within the Date range appear

Test Case 3: Display Seminar Detail(Seminar Organizer, Speaker, Abstract, Venue, Venue capacity, Date&Time, Number of Registered Attendees)		
TC03-01	View the Seminar detail(Organizer login)	All the Seminar details are shown clearly
TC03-02	View the Seminar detail(Speaker login)	All the Seminar details are shown clearly
TC03-03	View the Seminar detail(Attendee view)	All the Seminar details are shown clearly
TC03-04	View the Seminar detail(Admin login)	All the Seminar details are shown clearly
TC03-05	View the Seminar detail(Host login)	All the Seminar details are shown clearly
Test Case 4: Add Seminar		
TC04-01	Type Seminar Title	The Seminar Title is set
TC04-02	Type Seminar Description	The Seminar Description is set
TC04-03-01	Pick reasonable Seminar start Date and end Date	The Date is set
TC04-03-02	Pick unreasonable Seminar start Date and end Date	The Date is not set
TC04-04-01	Pick Seminar Organizer	Seminar Organizer is set
TC04-05-01	Pick Seminar Speaker	Seminar Speaker is set
TC04-05-02	Keep Seminar Speaker empty	No Speaker
TC04-06	Add Seminar Attendee	Attendee is added to the

		list
Test Case 5: Edit Seminar		
TC05-01	Change Seminar Title	Seminar title is changed
TC05-02	Change Seminar Description	Description is changed
TC05-03-01	Re-pick reasonable Seminar start Date and end Date	Seminar date is changed
TC05-03-02	Re-pick unreasonable Seminar start Date and end Date	Seminar date not changed
TC05-04-01	Re-pick Seminar Organizer	Seminar organizer changed
TC05-05-01	Re-pick Seminar Speaker	Seminar speaker changed
TC05-05-02	Pick no Seminar Organizer	Seminar will have no Organizer
TC05-06	Add Seminar Attendee	Attendee is added to the list
TC05-07	Change Seminar Attendee detail	Attendee detail is changed
TC05-08	Delete Seminar Attendee	Attendee is removed from the list
Test Case 6: Delete Seminar		
TC06-01	Delete the Seminar	Seminar removed
TC06-02	Delete the Attendee record for this seminar	Attendee removed

Test Case 7: Print Attendee Name Tags		
TC07-01	Print Attendee name tags in a file	Name Tags are opened in a pdf to print
Test Case 8: Add attendee to seminar		
TC08-01-01	Add Attendee to Seminar with name, phone number, email	Attendee added
TC08-01-02	Add Attendee to Seminar only with name	Attendee is not added
TC08-01-03	Add Attendee to Seminar only with phone number	Attendee is not added
TC08-01-04	Add Attendee to Seminar only with email	Attendee is not added
TC08-02	Add attendee with invalid information	Attendee is not added
TC08-03	Add an existing Attendee	Error, Attendee exists
Test Case 9: Change attendee		
TC09-01	Change Attendee name	Name changed
TC09-02	Change Attendee phone number	Phone number changed
TC09-03	Change Attendee's email	Email changed
TC09-04	Change Attendee with invalid information	Change unsuccessful
TC09-05	Add an existing Attendee	Error, Attendee exists
Test Case 10: Delete attendee		

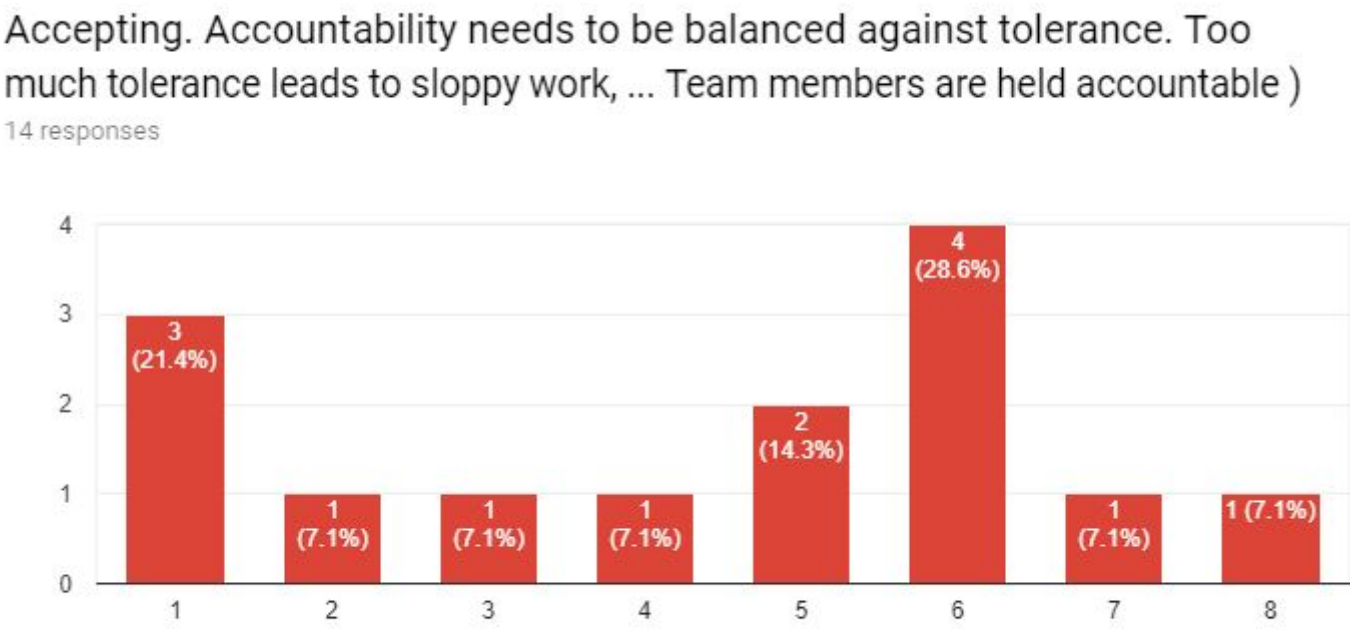
TC10-01	Delete the selected Attendee	Attendee removed
Test Case 11: Display User List		
TC11-01	View the summary list of Users	User list displayed
Test Case 12: Add user		
TC12-01	Add a user as an Organizer	User added as organizer
TC12-02	Add a user as a Speaker	User added as speaker
Test Case 13: Delete user		
TC13-01	Delete a User when he/she does not have an active Seminar	User deleted
TC13-02	Delete a User when he/she does have an active Seminar	Error, user is not deleted
Test Case 14: Change user		
TC14-01	Change User's basic information(name, biography, study area, education)	Basic information changed
TC14-02	Change User's Seminar list	List changed
Test Case 15: View the user details		
TC15-01	Open the User's details window	Details window shown

5. Appendix

Team Health

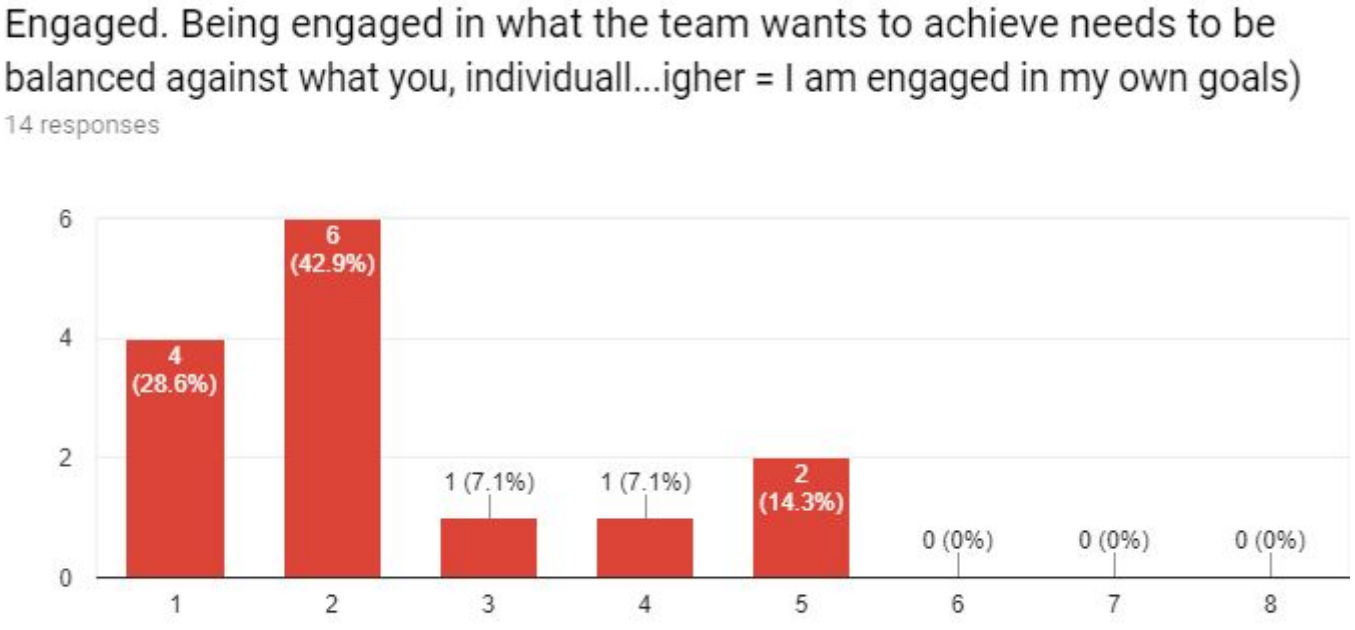
We measured our team health by seeing the results from the form that we use every week so see how we all are going with the group. This form measured the team's emotional wellbeing. Although there is a lot known about the importance of various emotional, as opposed to practical, aspects of teams there are very few useful measures. This was intended to find which measures are useful and informative. The intention was that teams would use the responses to inform discussions about what they should do.

Acceptance



Openess vs closedness

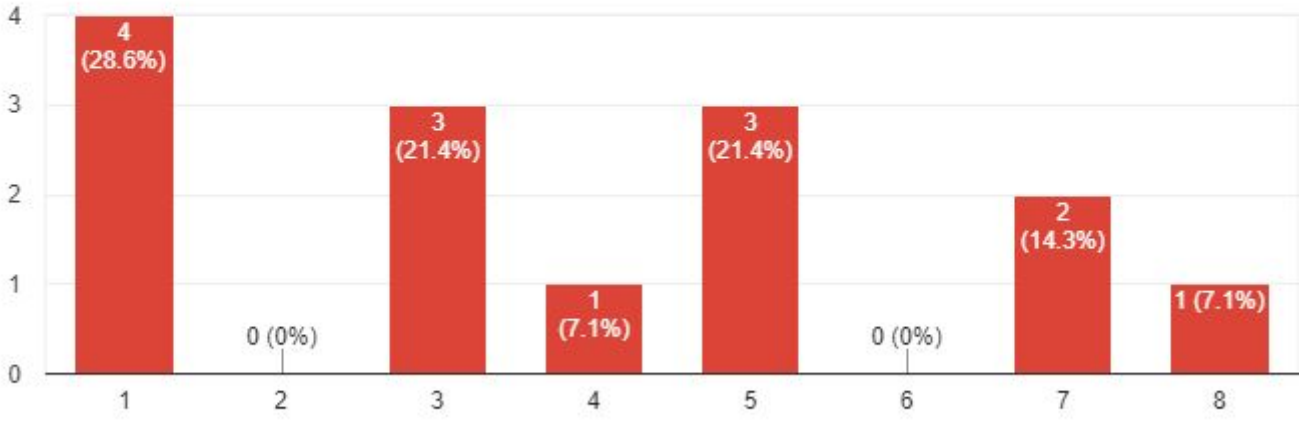
Engaged



Supported

Supported. Getting and giving help on some task needs to be balanced against completing your own tasks. Wh...her = I am accountable for my tasks)

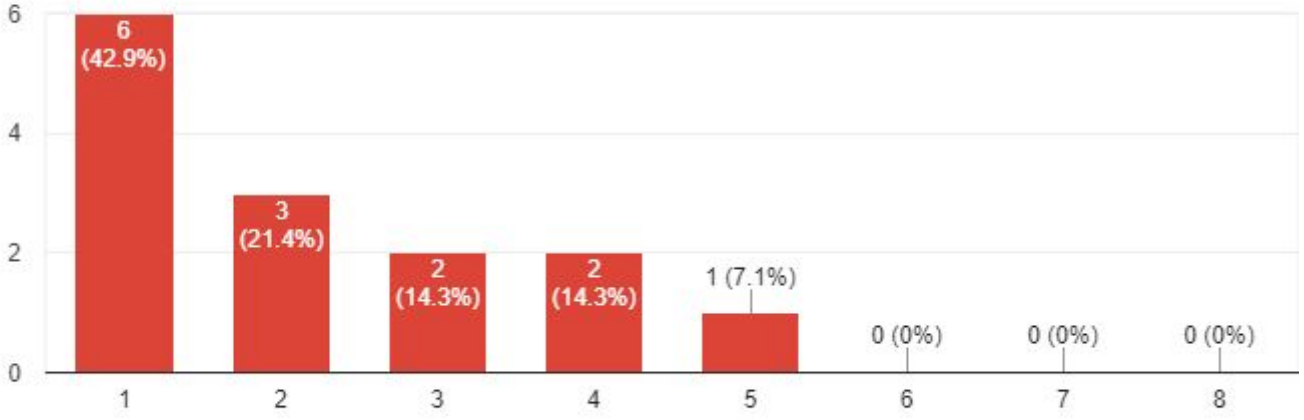
14 responses



Confident

Confident. Anything worthwhile will be challenging. Too little challenge give too little achievement but too much c... concerned that we might not succeed)

14 responses



Consensual. Too much consensus needs to be balanced against too much autonomy. Where is the team on this s... Collaborative, Higher = Autonomous)

14 responses