

Software Engineering Studio 3A/3B Team Charter (2021 Spring)

A team charter can be prepared for many reasons. One is to document the team's purpose and clearly define individual roles, responsibilities, and operating rules. Second, it can be used to establish procedures for both the team and agency management on communicating, reporting, and decision-making. It can be a blueprint for business acquisitions and it defines how the team is empowered to work, including assigning responsibility and authority. Last, it facilitates stakeholder buy-in by including key members in the decision-making process and helping to obtain their concurrence.

TEAM INFORMATION

Team Name	Friends
Product Owner	Alexa Donovan
Scrum Master	Variable
Scrum Team Members	George El-Zakhem, Jasmine Emanouel, Nuo Chen, Amrita Menon, Michelle Chen, Neeraj Gangrade Porwal, Nicholas Tsom, Jacob Willis

Every team member must briefly list personal knowledge, experience, expectation of the outcomes for the project, commitments (attendance of team activities, working hours and contributions) and roles with assigned tasks within the team. Further details about tasks can be listed in the task management/project timeline.

Name	Role	Sub-team	Knowledge / Experience	Expectation(s) and commitment(s)	Assigned tasks
Alexa	Project Manager	All	Javascript, React.js, Project Management, Project Architecture, Product Design Interned at Dematic, AirService. Software	To encourage and maintain progress of the team. To help develop and document the project.	See Timeline below

			Studios classes.		
George	Python Developer	Back-end	Internship in backend management using SQL. Proficient to a certain degree in JS and Python.	To help developer the functions which enable the project activity	See Timeline below
Jasmine	Quality Assurance manager	Front-end	Internships on both front-end and backend. Most front-end experience in react.	To help develop the user interface aspects of the project	See Timeline below
Nuo	Python Developer	Back-end	-pending-	To help developer the functions which enable the project activity	See Timeline below
Amrita	React developer	Front-end	Internship, utilising SQL, Java and python. Data science experience.	To research relevant data on various crisis events. To develop appropriate statistics, charts and other documentation required to help monitor and aid in said crisis.	See Timeline below
Michelle	React developer	Front-end	Graduate Developer <ul style="list-style-type: none"> • Java • Kotlin • Spring/API development • Corda • Postgres 	To help developer the functions which enable the project activity	See Timeline below

Jacob	Python developer	Back-end			See Timeline below
Nicholas	Security Specialist	Back-end			See Timeline below
Neeraj	Machine Learning/ AI specialist	Back-end			See Timeline below

PROJECT

Project scope and benefit

(State the scope, mission, milestones, and/or objectives in this project; list the expected benefits to the society.)

The purpose of this project is to provide a centralised emergency response app. This app addresses two key clients: civilians (general population) and authorities (trained individuals who are capable of handling a crisis).

Civilians are able to use this app to view nearby crisis, track people they know during the crisis, communicate with each other and the authorities, view safe and dangerous areas on a map, and view relevant procedures and information on crisis' near them.

Authorities can use this information to track people and resources during a crisis, view and communicate with civilians in need and view procedures and information about a crisis at hand.

This app will enable better organisation and awareness during crisis events (such as bushfire, drought, shark attacks, etc). It allows for better coordination of various groups trying to address a crisis event and will help minimise panic from civilians. Additionally, this project is designed to

work at small, medium or large sized events meaning it is flexible enough to use in a variety of situations.

Targeted users

(Identify the targeted users/customers; list users' needs and/or pain points clearly)

Emergency Services

Crisis management

Track people in need

Alerts public of health risks

Easy to use

Organizations that deal with small, medium and large scale crisis (e.g. charities)

Same as emergency services

Civilians

Personal information is secure

Can track friends and family at risk

Can summon aid when at risk

Can find safe areas

Can avoid dangerous areas

Easy to use

Pain Points

Civilian and authority needs do not always line up

Need to consider long term use for civilians, short-term (duration of crisis) use for authorities

Authorities all have different needs and responsibilities. Need to distill common ones

Environmental affects could reduce effectiveness of app (e.g. lack of connectivity)

Field or background research

(Complete a survey and comparison for the existing or similar solutions; list the advantages and drawbacks of the existing or similar solutions; list the superior milestones/functions/ideas in your project)

Service NSW app

Pros	Cons
efficient check-in	Design is to better enable government services, not crisis management
Neat design	Only caters to civilians

Gaurdly

Pros	Cons
Very nice UI	Complicated setup
Great geofence capability	Messaging is not very neat
Ease of use	doesn't consider large scale and/or outside city events

Great alert system, but isn't as applicable once separated from your support system such as when travelling or outside of cities.

Disaster alert

Pros	Cons
Good at informing of crisis	<ul style="list-style-type: none">• Scope is too broad• Only offers information
Nice map design	

Our map can be considered superior on a more personal basis. The added alert system allows us to compete with this functionality at a local level.

Life360

Pros	Cons
------	------

Very accurate tracking	Battery intensive
Good map design	doesn't offer additional functionality
Messaging functionality	Can only see people who you have the contact for. No broadcast capability
Text integration	

Acceptance testing for product

Identify and describe how and when the product will be tested over the course of the semester

Contributions will require testing from both the developer who submitted the contribution as well as an additional person before being merged into the develop branch of the repository.

Additionally, every 2nd week we will be running the project to address inconsistencies and issues. During the last 2 sprints of the project there will also be dedicated testers combing through the project as a whole to ensure an ideal standard has been met and to help address any final bugs and issues. For a function to be considered done it must pass the 'testing' process mentioned in the below *Functionality Composition* section and match to the 'definition of done' description.

Front-end testing will require quality assurance tests to be passed which will be managed using the software Chai to ensure no new changes have prevented any active functions. Back-end testing will also include regular penetration testing after the integration of the SHA-1 encryption on the database.

Definition of done for product as well as single function

Describe a set of conditions that allow your team to assess the product's completion for the semester

This product will be consider done if all functionality from the previous semester retains its quality, and the new epics of Messaging, Chatbot, Geolocation, Security and Quality Assurance pass their acceptance testing and are successfully integrated into the product.

Messaging will require peer-to-peer communication to work in real time with a lag of less than 4 seconds. It will also require integration with the Chatbot epic.

The Chatbot will be able to successfully gather information from a user with less than 2 'clarifications' required to create a new Event. It should also record and forward the discussion to a relevant authority upon conclusion of the talk.

Geolocation will allow for users to view all Events in reference to themselves on a map. It should also record the tracking data of the user with a deviation of less than 5 meters. Users should also be able to view other users locations.

Security should prevent basic automated bots from being able to access the system and also prevent any penetration testing attempts to fail.

Finally, Quality Assurance should be able to automatically determine whether changes have caused a break to the platform or to any of its functionality. This should be to the same standard as a manual test of the system.

Functionality composition

Function / Task	Priority	Difficulty	Completeness	Member(s)	Testing	Definition of Done	Resources
Event scraping							
Document 3A							
Chai dependencies							
Tensorflow dependencies							
Chat list UI							
Chat list call function							

- Commented [YW1]:** Please list every single function clearly
- Commented [YW3]:** Please list the members and members' roles
- Commented [YW4]:** Please describe the plan or method for testing
- Commented [YW6]:** List the required library, API, language, package, connected function(s), connected database and etc.
- Commented [YW2]:** Please scale the level between 1 - 5
1 -> Lowest
5 -> Highest
- Commented [YW5]:** Please define the meaning of done for every task and/or function
- To save space, you can number the different methods (testing and definition of done) and write the detail in above sections.

User story map

List user stories and a user story map based on the functions above

INVITE LINK:

<https://trello.com/invite/b/On7wNHBD/e0726bf08dae02f4e9ff25180567bb2a/user-stories-3b>

Project scale and sprint planning

Sprint period, milestone(s), deliverable and funding cost	<p>Sprint 1: week 2 – week 3 Deliverable: Onboarding complete, SES3A work documented, Help page, Resource page</p> <p>Sprint 2: week 4 – week 5</p>
--	---

- Commented [YW7]:** Please request the reasonable funding based on the estimated difficulty/complexity/novelty of your project.

	<p><i>Deliverable: Messaging (chat lists), Geolocation (display Events), Chatbot (dependencies and actions), Security (SHA-1), Quality Assurance (dependencies and build)</i></p> <p>Sprint 3: <i>week 6 – stuvac</i> Milestone: <i>Quality Assurance completed</i> Deliverable: <i>Messaging (create Chatbot chat), Chatbot (begin training, create Event function), Security (penetration testing, Google authenticator), Geolocation (store user location, display user location)</i></p> <p>Sprint 4: <i>week 8 – week 9</i> Milestone: <i>Messaging completed (pending Chatbot)</i> Deliverable: <i>Chatbot (ensure fully integrated in frontend, training), Geolocate (geofence alerts on Events), Security (penetration testing), have settings display Event controls for Admin</i></p> <p>Sprint 5: <i>Week 10 – week 12</i> Milestone: <i>Chatbot complete, Geolocate complete, Security complete</i> Deliverable: <i>bug fixes, video demo, presentation slides, UI/UX refinement as required</i></p>
Weekly meeting time (extra meeting is highly welcome)	<p>Thursday 7pm – React meeting – Michelle, Amrita, Alexa</p> <p>Friday 3pm – Quality Assurance meeting – Jasmine, Alexa</p> <p>Saturday 10am – Backend meeting – Nuo, Neeraj, Jacob, Nick, George, Alexa</p>

Timeline

Detail timeline and task assignment

SEE SPRINT PLANNING

Task assignment is expected to follow the below distribution:

Michelle, Amrita – front-end tasks as required mostly centred around UI development

Jasmine – Chai and Quality Assurance pieces as well as Messaging support.

George, Nuo, Jacob – python and testing tasks as required

Commented [YW8]: Please list the task and weekly schedule clearly.
Please use the spreadsheet to arrange the timeline as well as assign the tasks.

Nicholas – Security tasks

Neeraj – Chatbot and ML/AI tasks

Alexa – project management tasks (scoping, documentation, planning) as well as React tasks as required.

Team operations

(Describe team operational plans, for example, frequency of meeting, meeting preparation, progress report, attendance of meeting, team's decision-making processes, plans to establish ground or operating rules, or team activities, etc.)

SPRINT MEETING

Each monday is a regular weekly meeting. Discussion points to cover in this meeting is progress from the week, mark off progress from the sprint trello (link provided above under 'User Story Map'), group testing of the project, update 'design thinking' documentation, and assign work for the week.

Additionally, there are secondary sub-team meetings scheduled to ensure work is being completed at a good pace over the course of the week and that any issues can be addressed early on.

It is the responsibility of the individual that relevant materials are brought to/made available for the meetings based upon their role.

VARIABLE MEETINGS

From a week-to-week basis sub-teams/individuals may request an additional meeting to ensure velocity is maintained and to address issues. The content of these meetings will vary based on the requirements of that week. These meetings are non-compulsory and are not expected to be required with any frequency. Minimal note-taking is required for these meetings.

DECISION MAKING PROCESS

All decisions are made on a team basis. An issue may be raised at any time and will be further discussed during the full team meeting. Solutions may be raised to issues at this time and in the event of multiple options a compromise will be found or existing solutions will be voted upon.

OPERATING RULES

Submitting Code

All developers must create a branch to work on for their individual contributions. They must then raise a Peer Review (PR) after testing their work and have a second member of the team test and approve the content before squashing the branch and merging into the master branch. In the event of conflicts, the merge can be delayed until a group meeting (either relevant sub-team or

full team) where the members can work through the conflicts to ensure nothing important is lost. The original developer must then test the master branch to ensure no issues.

The naming conventions of the branch should <feature name>_<task> with the *feature name* matching to a card in Feature Progress trello and *task* being included to describe what part of the feature the branch addresses. If the branch fully addresses the feature the *task* can be set as *full*.

Submitting Documentation

For the purpose of the weekly journals a documentation git has been created to track non-code contributions to the project. Each topic (e.g. wireframes, crisis research, user stories, etc) will exist on a separate branch to enable ease of browsing. Each week before the monday meeting members must submit their progress.

To submit, documentation must be converted to a pdf and saved to the relevant branch. For the commit name follow the convention <document name>_<status> with the *document name* being a relevant and intuitive name and the *status* being *in progress* or *done* based on current stage.

RISK CONTROL

List the potential risks and mitigations/solutions during the project (see an example below)

Risk summary	Risk detail	Impact	Probability	Mitigation
Team member is overworked	Due to the busy schedules of the team, a member might be overloaded with work for one sprint.	The system might be delivered late if tasks are of a high priority and might deliver a low-quality product.	4	The team communicates their workload at weekly team meetings, and voices when they need assistance.
Team members are working in different time-zones	Team members working in different time-zones results in meetings that require members working in odd hours. Sometimes members may not be available for crucial meetings that could hinder the group's performance	Member's working in different time-zones will have difficulty in collaborating with other team members in completing milestone objectives. This in turn will slowly stunt the milestone progressions and increase difficulty relaying information to other members	2	The team will need to make equal sacrifices in working odd hours so that milestone targets can be met successfully.
Members cannot	Members not making team	Members not participating in team	3	The project leader would need to be able to take sprint notes between each

always make meetings	meetings could decrease the overall productivity of the team in reaching sprint deadlines.	meetings from unforeseen circumstances could result in miscommunication between members. Members could double up on completing tasks that were assigned to one another from the previously missed meeting.		<p>stand up meeting. The sprint notes, objectives and recordings would be posted on the chat for members to make references to them.</p> <p>The member that also missed the meeting should have a look at the sprint notes posted and recordings if available. The member should also contact the project at his/her time to be brought to speed with what is happening in more depth.</p>
Not all members have the same skill level/can work at the same pace	Members working collaboratively would naturally be at different levels. If a member lacks the technical skills to develop & implement a feature, they will slow down the efficiency of meeting the goal.	Members working in this collaborative environment may not be able to complete their tasks to meet the sprint deadlines because their partner is having difficulties implementing features. Sometimes the member may not be able to complete objective which would result in it being delayed and affecting other sprint deadlines.	3	<p>The team leader should be able to use the strengths of all team members to allocate them to tasks that best represent their skills. This will result in work being done at a higher quality and more efficiently. Members should also be cautious of their own abilities and if they lack the technical skills to complete sprint objective they should upskill immediately by seeking assistance or undergoing tutorials online.</p>

Other notes

BACKEND REPO - https://github.com/alexadonov/2021Spr_SES3B_Team4_Backend

FRONTEND REPO - https://github.com/jasmine-nahrain/2021_SES3A_Team4_Frontend

DOCUMENTATION REPO - https://github.com/alexadonov/2021Spr_SES3B_Team4_Documentation

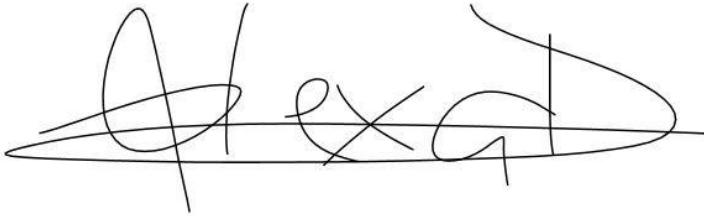
Signature of team members:

Name: Amrita Menon

Signature: A.M

Name: Alexa Donovan

Signature:

A handwritten signature in black ink that reads "Alexa". The letters are connected in a cursive style, with a large loop for the 'A' and a long horizontal stroke at the bottom.

Name: Jasmine Emanouel

Signature:

A handwritten signature in black ink that reads "Jasmine". The letters are connected in a cursive style, with a large loop for the 'J' and a long horizontal stroke at the bottom.

Name: Michelle Chen

Signature:

A handwritten signature in black ink that reads "Michelle". The letters are connected in a cursive style, with a large loop for the 'M' and a long horizontal stroke at the bottom.

Name: Nicholas Tsom

Signature:

Nuo Chen

Name: Nuo Chen

Signature:

Nuo

Name: Jacob Willis

Signature:

J Willis

Name: George El-Zarkhem

Signature:

A stylized, handwritten signature in black ink, consisting of several loops and a long horizontal stroke at the bottom.

Name: Neeraj Porwal

Signature:

A handwritten signature in black ink, appearing to be the letters 'n.g' in a cursive style.