uc3m | Universidad Carlos III de Madrid
Grupo

Software Development Project Management

Date: 02/25/2022 Group: 5

Carelynn Tsai
Clayton Rooke
Cooper Ang
Emily Wang
Erin Ly
Katherine Inglis
Karanbir Singh

Client: TD Bank

Table of Contents

Tables index

No se encuentran elementos de tabla de ilustraciones.

Figures index

No se encuentran elementos de tabla de ilustraciones.

CREATIVE IDEA DESCRIPTION

Workers have gotten consistently more productive as time goes on, a trend which has been clearly observed since the 1950s as a result of greater automation, education and other technological and social advancements. With these increases in productivity, employers have much higher expectations from workers every year; whether it be assigning more tasks, increasing the significance of projects worked on or other challenging activities. All of this has increased with no changes to the necessities of having an out-of-work life; balancing all of these factors, it's clear that worker burnout is getting much more common. Burnout affects workers negatively in many ways, from mental health impacts, to impacts on one's social life or commitments. These negative effects not only impact the individual personally, but are felt by companies as their employees' productivity decreases or they face greater resignation numbers. These issues are universal across nearly all workplaces. This is a growing issue, and the market size of a solution to reducing burnout is large and can be used at workplaces of all sizes in all locations around the world. Reducing employee burnout, even by a small amount, can result in a big positive impact on a company's health.

There are various causes and contributing factors to burnout that vary across individuals. Overworking is one of the most common causes of burnout. Employees may feel the pressure to be constantly working, making it hard to take breaks. Taking a break may seem unnecessary and unproductive, leading to feelings of guilt. Another factor of burnout is the absence of community and social-support. Individuals who overwork may not feel motivated to socialize, which can provoke or worsen their symptoms of burnout. Feeling unsupported and lonely during times of stress can have serious consequences on individuals' mental health which not only impact performance in the workplace, but also their day-to-day functions.

Our product proposal, Break Buddies, encourages taking breaks while connecting with others by automating the decisions and removing any barriers. It makes it simple for people to take breaks during the work day by automatically organizing them together in convenient time slots. Based on each participant's interests and preferences, it provides content in the form of video or audio to facilitate the activity. Having a set plan allows participants to get started right away, rather than spending time figuring out what to do.

Creating a solution which engages groups generates additional value in two ways. Firstly, it improves the possibility for social relationships at the workplace, which in turn improves mental health of individuals, improves the work environment, and reduces burnout. Another benefit to focusing on a group-focused activity is that it allows a larger sample size to determine which activities, times, group organizations and other variables work best to reduce burnout, and increase other positive outcomes.

The target audience for this solution is large organizations with in-person office spaces. In these organizations, it can be difficult to engage people individually, especially when certain negative work cultures have had time and a structural ability to fester.

The functionality of our solution is as follows: Break Buddies will schedule employee breaks automatically with some customizability; integrating with member's enterprise calendar systems; as well as the company's meeting room booking systems. It will automatically choose the perfect time to schedule an activity based on the calendars of each individual participant. The system will help with creating groups based on availability, interests, personal invitations, and other factors. To encourage attendance, the group size will

be kept small, and of a customizable higher frequency to encourage frequent breaks. Based on the parameters of the activity chosen, and the size of the group, the software will present some suitable meeting areas which will be automatically selected. The activities, which may include content watching and listening, will be sourced from third parties; and users can suggest content or activities to add to the platform.

This will be a web application so it can be easily accessible on employee devices. Each employee will have an account and profile. Users can adjust their preferences as mentioned above; and certain activities will be able to be hosted on the platform such as video, audio, and web games. Users will set up their accounts using their existing organizational credentials to be connected to calendar and other services. They will then be able to set up their profile, with activities that they like, customize availability, frequency of activities, group sizes, and invite members to groups. The user will be informed of any groups or activities they have been added or invited to. They will also be able to choose whether they want to join, or if they don't, why not, to help curate their experience. Upon completion of a group activity, the users will be able to provide feedback to help influence future recommendations. The enterprise admin will be able to see employee trends; to find out which activities or other dynamics are popular, and to see which activities are working well to reduce burnout. For enterprise administrators, care is taken at this stage, to ensure that individual worker privacy is respected, so individual feedback will be anonymized.

The web app itself will be linked to a web server and database with user authentication details and calendars, which will be set up to sync with the existing organizational systems. Logged in users will be stored in tokens on the web application. For the planning of events that fit into multiple user's calendars, that simultaneously align with their interests and preferences, an algorithm will be used to maximize the chance of a match, which will dynamically update based on the user's response. The web application will have a standard login screen, after which the user will be greeted by menus to lead to any of the features which they need. Notifications and booked events are put first and foremost to ensure users remember them. Preferences will be easily accessible for users to determine what works best for them and make changes to the settings once they gain comfort with the system. Once an activity starts, if the activity is supported within the app; i.e. movies, YouTube videos, certain web games; users will be able to interact with it from directly within the app; for activities in non-digital spaces; i.e. meditation, exercise, and nature walks; some helpful information, scheduling or instructions will be provided in the virtual activity space. Admins will be able to select permissible activities across the organization, modify room sizes and general organizational dynamics, and view feedback and user trends.

1.General data of the company offering the project

Name: SiestAI
Acronym: Non-applicable

Description: Software company building products to make the workplace a happier and more efficient place.

Mission: Our mission is to build software products focused on the enterprise employee experience. Our long-term vision is to use software technologies with the addition of AI for the betterment of the workplace. We are aimed to help with the digital transformations of the workplace by infusing technology into the everyday experience of the knowledge worker. Our software solutions and consulting work cover use cases across many

industries such as information technology, financial services, natural resources, and governmental work.

2.Definitions and acronyms

TD Bank	Toronto Dominion Bank

3.Initial offer and budget

Our company is SiestAI, a software consultant company building products for the workplace, focused on improving the employee experience. Our mission is to make the workplace healthier, happier, and more efficient.

TD Bank is the largest commercial bank in Canada, offering a range of financial services & products to 15 million customers, with 90,000 employees worldwide. One of TD's guiding principles is to "Be an Extraordinary Place to Work" and we can help TD achieve this.

3.1 Offer

SiestAI will develop the application, Break Buddies, to be used in TD offices around the world. Break Buddies is a web platform that integrates with corporate calendars to schedule tailored social and individual activities amongst large groups, specifically within large-sized organizations with in-person function. The mission is to encourage work-life balance and social interaction in the office by collectively simplifying break scheduling. This will help boost employee productivity and morale, as well as promote healthier habits to reduce the rate of burnout.

There are currently no applications that offer the same functionality and services all-in-one platform. Some existing software with similar aspects to Break Buddies include StandUp (a work break timer) and StrideKick (an app that allows people to compete in fitness challenges). StandUp is designed for the individual and works manually, compared to Break Buddies which schedules breaks according to a group of employees' calendar schedules. StrideKick encourages fitness through the asynchronous challenges amongst friends or co-workers, whereas Break Buddies focuses on synchronous group activity sessions. Break Buddies will be the 1st platform designed for the enterprise and office settings.

Other competing software development companies, such as TribalScale, do not have the same expertise and experience as SiestaAI in developing applications for the workplace and improving the employee experience. SiestAI has 8 years of experience delivering 15+ projects in this space. SiestAI follows iterative and agile practices to ensure a human-centered design.

For our Action Plan, the project is expected to take nine months to complete. Based on the complexity of the project, it will require a project manager, two software engineers,

and three programmers. The length of the development phase is estimated using the drafted use case model.

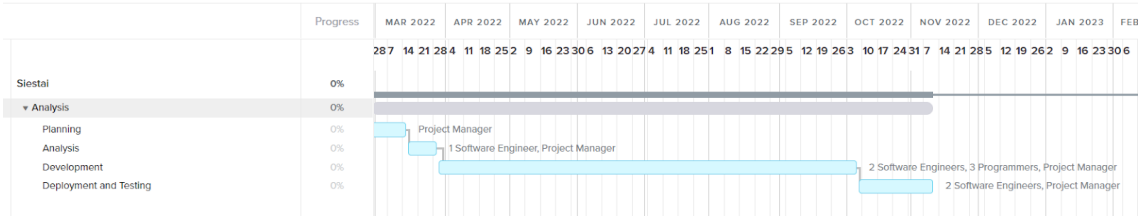


Figure 1: Action Plan

Going beyond the initial proposal of deploying Break Buddies for select TD Bank offices, we hope to create a long-term partnership. This includes monitoring the usage and performing necessary maintenance on the platform, implementing feature requests and fixes from feedback from TD employees, and iterating on features based on successful engagement. SiestAI hopes to further scale Break Buddies company-wide across TD Bank's various sectors.

3.2 Budget

Estimating the budget for the project started with an action plan containing the project phase timeframes as well as which team members will be involved in each phase. Table 1 describes the roles and pay for each employee. Amounts were taken from current industry evaluations in early 2022.

Table 1:Required roles for the project and the salary per month

Role	Pay per Person per Month (€)
Software Engineer	3,250
Programmer	2,760
Project Manager	3,300

To estimate the length of the development phase, we evaluated the complexity of each use case in the drafted use case model, assigning a number of weeks to each use case. The use case model is shown in Figure 2.

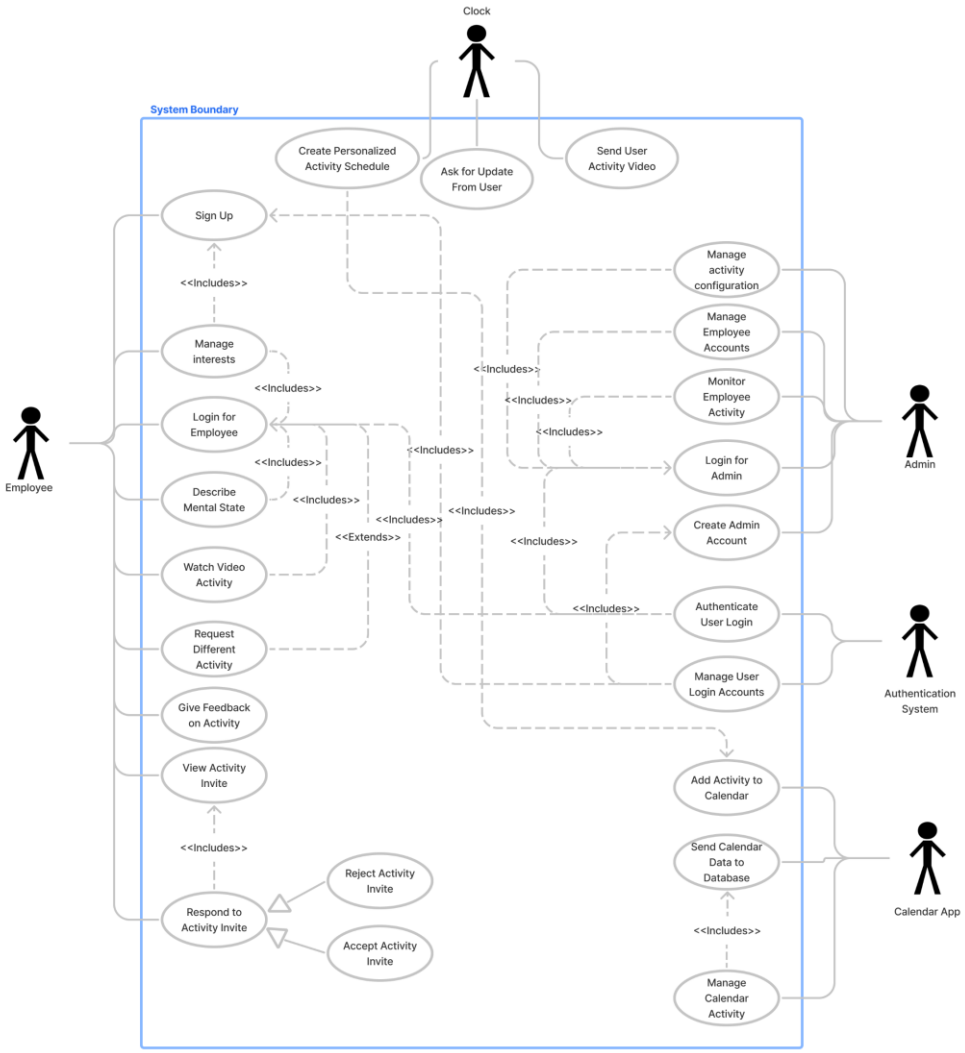


Figure 2: Draft of the use case model for Break Buddies

The majority of the budget is allocated to pay for the employees who will work on the project. Smaller amounts of the budget are also reserved for equipment and expenses. Our firm takes a 20% profit margin, and another 20% is allocated for risk. The scale and novelty of the project were the motivation for choosing a large risk margin. The breakdown of the budget is included in table 2 on the following page.

Table 2: Proposed budget for Break Buddies

Salary of the teamwork:	
Planning	€1,650.00
Analysis	€3,275.00
Development	€115,260.00
Deployment and Testing	€9,800.00
Total (for teamwork)	€129,985.00

Additional costs (VAT inclusive on purchases):	
Computer Equipment (5 MacBook's)	€7,500.00
Software (Cloud Hosting)	€5,000.00
Consumables	€500.00
Travel and Expenses (Business meetings for project manager)	€600.00
Total (for additional costs)	€13,600.00
Profit margin and risk:	
Total (salaries and additional costs):	€143,585.00
Profit Margin (20%)	€28,717.00
Risk (20%)	€28,717.00
Taxes:	
Subtotal	€201,019.00
VAT (21%)	€42,213.99
Grand Total	**€243,232.99**

Explanation of the budget:
The understanding of Spanish tax is that VAT is levied on the purchase of goods (additional costs), which is inclusive in the above table. Salaries are not taxed when paid to the employees. The final sub total of the service, inclusive of all the costs, are taxed by VAT before the client pays for the goods (21% of €201,019.00).

4.Software Configuration Management Plan

INTRODUCTION

4.1 Purpose of the Plan

The Plan detailed below is aimed at both the development staff and the management team. The aim is to make the project sufficiently robust to collect information about the state of the product and to make a change. The changes are especially delicate in this one, since there are elements that require special attention and care when modifying them.
It is therefore intended to document each baseline and each change made as indicated below when detailing configuration management activities.

4.2 Scope

This SCM plan will apply to the project Break Buddies, a web-based break planner developed by SiestAI to be used by TD Bank

4.3 Definitions and Acronyms

The following are the acronyms used in this Configuration Management Plan.

SCM	Software Configuration Management
TD Bank	Toronto Dominion Bank
AWS	Amazon Web Services
CE	Configuration Element
UI	User Interface
API	Application Programming Interface
UML	Unified Modeling Language

4.4 References

Larman, C. (2005). *Applying UML and patterns : An introduction to object-oriented analysis and design and iterative development* (3rd ed.). Upper Saddle River, N.J.: Prentice Hall PTR, c2005.

MANAGEMENT SPECIFICATIONS

This section identifies the coordination and management tasks that will be necessary to carry out the SCM.

4.5 Organization

There must be permanent and direct contact between the development staff and the change control committee, so that delays in the processing of a change are as short as possible, so that both improvement and correction processes are not tedious work.
Both the change control committee and the other development staff should pay special attention to the points where it has been stipulated that baselines will be established within the development. For more information see the section on Definition and Establishment of Baselines.

4.6 Responsibilities

Change control committee: Clayton Rooke, Emily Wang

Responsible for SCM: Carelynn Tsai, Karanbir Singh

Librarian: Cooper Ang

Rest of the development staff: Erin Ly, Katherine Inglis

4.7 Applicable policies, directives and procedures

The applicable procedures are described in the section: " ".

CONFIGURATION MANAGEMENT ACTIVITIES

The following is a description of the SCM activities that will be carried out during the development of this project.

4.8 Configuration Identification

4.8.1 The preliminary product hierarchy is established

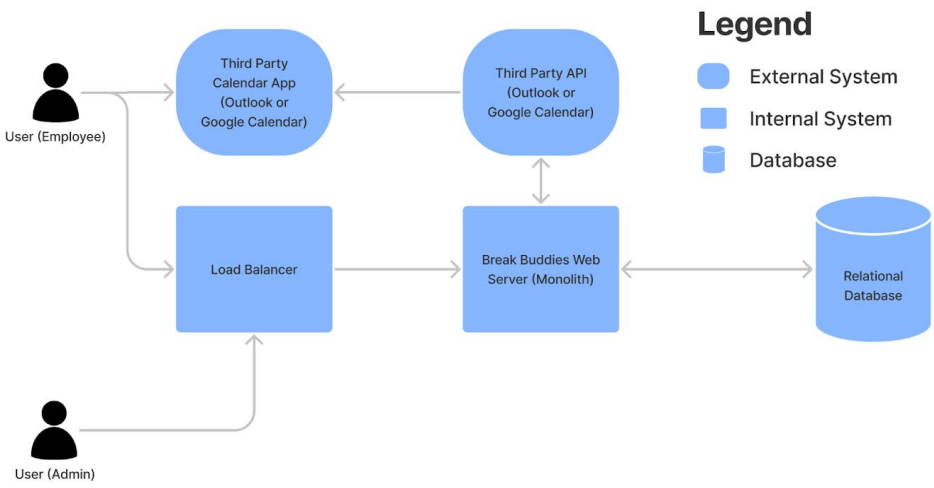


Figure 3: General System Structure

4.8.2 Selection of the configuration elements

The configuration elements are classified into three broad categories, as follows.

Initial Phase (INT)
These configuration elements outline the high-level overview of the project plan.

Planning and Requirements Specification (PRS)
These configuration elements guide the organization of requirements and use cases.

Construction (CON)
These configuration elements outline the design and analysis in detail and will directly contribute to the development of the product.

4.8.3 Selection of the identification scheme

Each CE will have a unique ID which includes a 3-letter prefix for the type of CE, a 2-letter descriptive code for the specific CE, followed by 1 number indicating iteration. Established types of CEs and their corresponding 3-letter prefixes include Development (DEV), Documentation (DOC), and Management (PRS). The CEs and corresponding identifiers are outlined in table 3.

CE Schema:

- CE Code (eg. CON-SC-2 is Development: Source Code)
- Full name of CE
- Description
- Date of Creation
- Project to which it belongs
- Baseline to which it belongs
- Permissions Level Access (All, Developers, Owners)

Table 3: CE Schema

CE Code	Name and Description	Date of Creation	Baseline	Permissions Level Access
INT-OB-1	Offer and Budget			
PRS-QP-1	Quality Plan			
PRS-SP-1	SCM Plan			
PRS-PR-1	SCM Plan Review			
INT-UM-1	Use Case Model			
INT-UR-1	Use Cases Model Review			
PRS-ES-1	Estimation			
PRS-ER-1	Estimation Review			
PRS-SC-1	Schedule			
PRS-SR-1	Schedule Review			
INT-FA-1	Feasibility analysis			
INT-FR-1	Feasibility review			
INT-UC-1	Prioritization of use cases			

INT-RU-1	Prioritization of use cases review.			
INT-UD-1	Definition of high-level use cases			
INT-RD-1	Definition of high-level use cases review			
CON-UE-1	Use cases in extended format (iteration 1)			
CON-UR-1	Use cases in extended format review (iteration 1)			
CON-CM-1	Conceptual model (iteration 1)			
CON-CR-1	Conceptual model review (iteration 1)			
CON-OC-1	Operation Contracts (iteration 1)			
CON-OR-1	Operation Contracts review (iteration 1)			
CON-CD-1	Class diagram (iteration 1)			
CON-DR-1	Class diagram review (iteration 1)			
CON-SD-1	Sequence diagrams (iteration 1)			
CON-SR-1	Sequence diagrams review (iteration 1)			
CON-TS-1	Transition states diagram (iteration 1)			
CON-TR-1	Transition states diagrams review (iteration 1)			
CON-UE-2	Use cases in extended format (iteration 2)			

CON-UR-2	Use cases in extended format review (iteration 2)			
CON-CM-2	Conceptual model (iteration 2)			
CON-CR-2	Conceptual model review (iteration 2)			
CON-OC-2	Operation Contracts (iteration 2)			
CON-OR-2	Operation Contracts review (iteration 2)			
CON-CD-2	Class diagram (iteration 2)			
CON-DR-2	Class diagram review (iteration 2)			
CON-SD-2	Sequence diagrams (iteration 2)			
CON-SR-2	Sequence diagrams review (iteration 2)			
CON-TS-2	Transition states diagram (iteration 2)			
CON-TR-2	Transition states diagrams review (iteration 2)			
CON-UE-3	Use cases in extended format (iteration 3)			
CON-UR-3	Use cases in extended format review (iteration 3)			
CON-CM-3	Conceptual model (iteration 3)			
CON-CR-3	Conceptual model review (iteration 3)			

CON-OC-3	Operation Contracts (iteration 3)			
CON-OR-3	Operation Contracts review (iteration 3)			
CON-CD-3	Class diagram (iteration 3)			
CON-CD-3	Class diagram review (iteration 3)			
CON-SD-3	Sequence diagrams (iteration 3)			
CON-SR-3	Sequence diagrams review (iteration 3)			
CON-TS-3	Transition states diagram (iteration 3)			
CON-TR-3	Transition states diagrams review (iteration 3)			

4.8.4 Definition of relationships

The relationships between CEs will be defined using the following table format. The possible relationships include dependence, derivation, succession, equivalence, variation, and composition. The table will be completed in the status account.

Table 4: Equivalent Relationships

CE Code	Location	Date

Table 5: Dependence Relationships

CE 1 Code	CE 2 Code	Date

Table 6: Derivation Relationships

CE 1 Code	CE 2 Code	Date

Table 7: Variant Relationships

CE code	Main Branch	Variants	Date

Table 8: Succession Relationships

CE code	Previous version	Next version	Date

Table 10: Composition Relationships

Parent CE Code	Child CEs	Date

4.8.5 Definition and establishment of baselines

The baselines will be defined using the following table format in Table 11. They are defined using the modified Craig Larman methodology, with a baseline for each iteration of each phase of the project. The construction iterations contain multiple baselines. Coding, Testing, and Installation are not included as baselines since they do not contain any CEs for this project. The contents of each baseline are included in the status account.

Table 11: Definition of the Baselines for the Project

Baseline Name	**Included CEs/Baselines**	**Closing Date**
Phase 0		
Planning and Requirements Specification		
Construction Phase Iteration 1		
Iteration 1 Analysis		
Iteration 1 Design		
Construction Phase Iteration 2		
Iteration 2 Analysis		
Iteration 2 Design		
Construction Phase Iteration 3		
Iteration 3 Analysis		
Iteration 3 Design		

4.8.6 Definition and establishment of software libraries

Table 12: Software Libraries Definition

Software Library	**Location**
Work (one per developer) To organize work libraries, a folder will be assigned to each developer to keep individual work separate and organized	siestAI/server/projects/break_buddies/work/{developer_name} Examples: siestAI/server/projects/break_buddies/work/cooperang siestAI/server/projects/break_buddies/work/emilywang
Integration (to integrate CEs) To organize integration libraries, a folder will be assigned to each CE	siestAI/server/projects/break_buddies/integration/{CE} Examples: siestAI/server/projects/break_buddies/integration/documentation
Support (to house the integrated, and therefore finished, CEs) To organize support libraries, a folder will be assigned to each CE	siestAI/server/projects/break_buddies/support/{CE} Examples: siestAI/server/projects/break_buddies/support/documentation
Production (to store the base lines) To organize production libraries, a folder will be assigned to each baseline	siestAI/server/projects/break_buddies/production/{baseline} Examples: siestAI/server/projects/break_buddies/production/phase_0/PRS
Master (client versions)	siestAI/server/projects/break_buddies/master/{client_version}

To organize master libraries, a folder will be assigned to each client version and release	Examples: siestAI/server/projects/break_buddies/master/client_version_1.1
Software or repository (other projects) To organize other project libraries, a new project folder will be assigned to each project	SiestAI/server/projects/{project_name} Examples: SiestAI/server/projects/other_project
Backup To organize backup libraries, a folder will be assigned to each backup version	siestAI/server/projects/break_buddies/backups/{backup_version} Examples: siestAI/server/projects/break_buddies/backups/backup_version1.1

4.9 Changes control

It is requested:

- Applicable change control procedure

The change control procedure will follow the IEEE 1042-87 standard:

1. Change request is made
2. Request is classified and registered
3. Request is evaluated by the Change Control Committee
4. Request is approved or rejected
5. If approved, change order is given and affected developers are notified
6. Change is made
7. Change is assessed and certified
8. Originator of change is notified

- Change Request Report Format

Request for Change: *Title of change request* XXX-XX *(CE Unique ID)* - YYYY-MM-DD *(Date)*
Requested by: *Person(s) requesting the change*
Reasoning:

Why is the change necessary? (1 or 2 sentences)
Description: *What specific part of the CE needs to be changed? How will the change be implemented? (2 or 3 sentences)*
Damage Estimate: Estimate of work hours required: XX Estimate of cost: €XX
Affected Areas: *List other affected CEs and affected teams.*
Approval: (To be filled in by Change Control Committee) YES/NO *Comments*

- Change Certification Report Format

Certification of Change: *Title of original change request* XXX-XX *(CE Unique ID)* - YYYY-MM-DD *(Date)*
Implemented by: *Person(s) who implemented the change*
Description: *How was the change implemented? Which CEs were affected? (2 or 3 sentences)*

4.10 Status account

The status account is contained in tables 13, 14, and 15.

Configuration Elements:

Table 13: Configuration Elements

CE Code	Name and Description	Date of Creation	Baseline	Permissions Level Access
INT-UM-1	Use case model: Analysis of a users'	2022-02-25	Planning and Requirements	Owners

	interactions with the system		Specification Phase	
INT-OB-1	Budget: Resources necessary to be allocated for project objectives	2022-02-25	Phase 0	Owners
INT-FA-1	Feasibility Analysis: Feasibility study and requirements specification	2022-03-14	Planning and Requirements Specification Phase	Owners
PRS-PR-1	SCM Plan Review	2022-03-22	Phase 0	Owners
INT-UR-1	Use Cases Model Review	2022-03-22	Phase 0	Owners
INT-FR-1	Feasibility review	2022-03-22	Phase 0	Owners
INT-UC-1	Prioritization of use cases	2022-03-28	Planning and Requirements Specification Phase	Owners
INT-RU-1	Prioritization of use cases review.	2022-03-28	Planning and Requirements Specification Phase	Owners
INT-UD-1	Definition of high-level use cases	2022-03-28	Planning and Requirements Specification Phase	Owners
INT-RD-1	Definition of high-level use cases review	2022-03-28	Planning and Requirements Specification Phase	Owners
PRS-ES-1	Estimation: Detailed estimation for use cases, actor weighting, technical complexity factors, and environmental factors	2022-04-05	Planning and Requirements Specification Phase	Owners
PRS-SC-1	Schedule: Detailed Gantt chart generated in MS Project	2022-04-05	Planning and Requirements Specification Phase	Owners
PRS-ER-1	Estimation Review: A holistic review was conducted of the estimation.	2022-04-05	Planning and Requirements Specification Phase	Owners
PRS-SR-1	Schedule Review: Schedule was - reviewed by	2022-04-05	Planning and Requirements	Owners

	different member of the team		Specification Phase	
CON-UE-1	Definitions of use cases in extended format (iteration 1)	2022-05-04	Construction Phase	Owners
CON-OC-1	Definition of operation contracts (iteration 1)	2022-05-04	Construction Phase	Owners
CON-CD-1	Construction of Class Diagrams (iteration 1)	2022-05-04	Construction Phase	Owners
CON-SD-1	Construction of Sequence Diagrams (iteration 1)	2022-05-04	Construction Phase	Owners
CON-TS-1	Construction of Transition states Diagrams (iteration 1)	2022-05-04	Construction Phase	Owners
CON-UE-2	Definitions of use cases in extended format (iteration 2)	2022-05-05	Construction Phase	Owners
CON-OC-2	Definition of operation contracts (iteration 2)	2022-05-05	Construction Phase	Owners
CON-CD-2	Construction of Class Diagrams (iteration 2)	2022-05-05	Construction Phase	Owners
CON-SD-2	Construction of Sequence Diagrams (iteration 2)	2022-05-05	Construction Phase	Owners
CON-TS-2	Construction of Transition states Diagrams (iteration 2)	2022-05-05	Construction Phase	Owners

Relationships:

Table 14: Dependence Relationships

CE 1 Code	CE 2 Code	Date
PRS-SC-1	PRS-BU-1	2022-03-07
PRS-ES-1	PRS-SC-1	2022-04-19
CON-UE-1	CON-OC-1	2022-05-04
CON-OC-1	CON-SD-1	2022-05-04
CON-UE-2	CON-OC-2	2022-05-05
CON-OC-2	CON-SD-2	2022-05-05

Table 15: Derivation Relationships

CE 1 Code	CE 2 Code	Date

INT-UM-1	PRS-BE	2022-03-07
INT-UM-1	INT-FA-1	2022-03-14
PRS-SP-1	PRS-QP-1	2022-03-07
INT-UM-1	INT-UD-1	2022-03-28
INT-UM-1	INT-UC-1	2022-03-28

Table 17: Succession Relationships

CE code	Previous version	Next version	Date
INT-UM	1	2	2022-03-22
INT-UM	2	3	2022-03-28
INT-OB-1	1	2	2022-02-25
INT-FA-1	1	2	2022-03-14
PRS-PR-1	1	2	2022-03-22
INT-UR-1	1	2	2022-03-22
INT-FR-1	1	2	2022-03-22
INT-UC-1	1	2	2022-03-28
INT-RU-1	1	2	2022-03-28
INT-UD-1	1	2	2022-03-28
INT-RD-1	1	2	2022-03-28
PRS-ES-1	1	2	2022-04-05
PRS-SC-1	1	2	2022-04-05
PRS-ER-1	1	2	2022-04-05
PRS-SR-1	1	2	2022-04-05
CON-UE-1	1	2	2022-05-04
CON-OC-1	1	2	2022-05-04
CON-CD-1	1	2	2022-05-04
CON-SD-1	1	2	2022-05-04
CON-TS-1	1	2	2022-05-04
CON-UE-2	1	2	2022-05-05
CON-OC-2	1	2	2022-05-05
CON-CD-2	1	2	2022-05-05
CON-SD-2	1	2	2022-05-05
CON-TS-2	1	2	2022-05-05

Baselines:

Table 18: Closed baselines for the project

Baseline Name	**Included CEs/Baselines**	**Closing Date**
Phase 0	INT-OB-1, PRS-QP-1, PRS-SP-1, PRS-PR-1, PRS-ES-1, PRS-SC-1, PRS-SR-1	2022-03-14
Planning and Requirements Specification Phase	INT-UC-1, INT-UM-1, INT-FA-1, INT-RU-1, INT-UD-1, INT-RD-1, PRS-ES-1, PRS-SC-1, PRS-ER-1,	2022-04-05

	PRS-SR-1, CON-TS-1, CON-UE-2, CON-OC-2	
Construction Phase	CON-UE-1, CON-OC-1, CON-CD-1, CON-SD-1, CON-CD-2, CON-SD-2, CON-TS-2, CON-UR-1, CON-CR-1, CON-OR-1, CON-DR-1, CON-SR-1, CON-TR-1, CON-UR-1, CON-TR-1, CON-UR-2, CON-CR-2, CON-OR-2, CON-DR-2, CON-SR-2, CON-TR-2, CON-UR-2, CON-TR-2	2022-05-05

4.11 Configuration auditing

Table 19 contains the history of completed reviews.

Table 19: Status of completed reviews of CEs

Review CE Code	Review Status	Date
PRS-PR-1	PASSED	2022-03-22
INT-UR-1	PASSED	2022-03-22
INT-FR-1	PASSED	2022-03-22
INT-RU-1	PASSED	2022-03-28
INT-RD-1	PASSED	2022-03-28
PRS-ER-1	PASSED	2022-04-05
PRS-SR-1	PASSED	2022-04-05
CON-UR-1	PASSED	2022-05-06
CON-CR-1	PASSED	2022-05-06
CON-OR-1	PASSED	2022-05-06
CON-DR-1	PASSED	2022-05-06
CON-SR-1	PASSED	2022-05-06
CON-TR-1	PASSED	2022-05-06
CON-UR-2	PASSED	2022-05-06
CON-CR-2	PASSED	2022-05-06
CON-OR-2	PASSED	2022-05-06
CON-DR-2	PASSED	2022-05-06
CON-SR-2	PASSED	2022-05-06
CON-TR-2	PASSED	2022-05-06

Included below is the history of changes to the CEs using the specified change request and certification formats.

Request for Change: Change Tax calculation to include profit margin and risk values INT-OB-1 - 2022-03-13
Requested by: Katherine Inglis

Reasoning: Based on feedback, the tax calculation and the resulting total of the budget had to be modified to follow the tax codes in Spain.
Description: The tax was levied on the profit margin and the risk calculation in addition to all the previous costs.
Damage Estimate: Estimate of work hours required: 1 hour Estimate of cost: €30
Affected Areas: *None*
Approval: (To be filled in by Change Control Committee) YES

Certification of Change: Change Tax calculation to include profit margin and risk values INT-OB-1 - 2022-03-13
Implemented by: Cooper Ang
Description: The budget was recalculated, the tax row was added below profit margin and risk and the total was updated. This affected the budget CE.

5. Quality Plan

CONTENT OF THE QUALITY ASSURANCE PLAN FOR THE INFORMATION SYSTEM

In the successive points of the document, the detailed tasks that are going to be carried out in the fulfilment of the Quality Assurance Plan will be exposed to check that the whole project fulfils the necessary quality criteria and that they have been considered as indispensable for the correct accomplishment of the project.

The revisions will be made as the project phases are completed until the final and complete design of the product is reached.

Those responsible for carrying out the revisions and accepting the validity of the products will be Clayton Rooke as Quality Manager and Katherine Inglis as Project Manager. In addition, all the members of the work team must carry out the revisions assigned by the Project Manager and communicate to the two people in charge of the Quality Assurance Plan in the event that any fault is found.

The following points of the document detail the specific reviews that will have to be carried out in compliance with the Quality Assurance Plan. The establishment of this quality assurance plan will begin in the System Feasibility Study and will be applied throughout the development of the software project (analysis, design, implementation...).

For each of the revisions, an Audit Report must be added that includes the approval or rejection of the revised product, indicating, if necessary, the causes for rejection of said product.

REVIEW OF THE SYSTEM'S FEASIBILITY STUDY

DOCUMENT REVIEW

Clayton Rooke, as Quality Manager, will confirm that the requirements have been specified in a structured way, with a precise and complete content, as established in the Quality Assurance Plan. Our Quality Manager will ensure that the requirements specification document offers the following features:

- Identification of absolutely all user requirements.
- Consistency between the content of the document and its objective.
- Each requirement describes the functionality that corresponds to it.
- Correspondence between the requirements of the document and the requirements obtained from the user, so the requirements specification is complete.
- Description of the requirements in clear, unambiguous language and therefore precise
- The feasibility study is self-descriptive, as its structure and content are described.
- A requirements traceability matrix shall be carried out to check that all user requirements have at least one software requirement associated with them and are thus present in the system design.

REVIEW OF USE CASES

REVISION OF THE USE CASE DIAGRAM

Use cases are a very important tool in the software development process and we use them to estimate activities before modeling or building a software development process.

With the use cases we have the functionalities and characteristics or basic requirements of the system. They are not based on any language so they are independent of them.

From the use cases, using the use case method, the size of the software will be estimated. The requirement to be able to use this tool is to define a use case model that represents well the domain of the problem to be addressed.

Clayton Rooke, as Quality Responsible, must carry out the revision of the Use Case Diagram, for this he must verify that the use case diagram complies with the following:

- The use case diagram describes the behaviour of the system, i.e. the complete functionality of the software project to be developed.
- The use case diagram includes all identified use cases representing all system functionalities.
- The use case diagram includes all the actors identified and involved in the system.
- The use case diagram includes all the dependencies and relationships between actors and use cases.
- The use case diagram complies with the graphic notation defined in UML modeling language.
- The use case model includes a glossary of terms that describes the terminology used.

REVIEW OF HIGH-LEVEL USE CASES

Clayton Rooke, as Quality Manager, must carry out the revision of the high level Use Cases, to do so, he must verify that they comply with the following

- The high-level use cases contain the name, actors, description and type of use case.

- Each use case describes how to achieve a single goal, that is, it describes a feature of the system.

- Each use case contains a textual description of the functionality associated with the appropriate level of detail, including ways in which the intended actors could work with the system. The description will use the language of the end user.

- The use cases do not describe internal system functionality, nor do they explain how it will be implemented. They do not include technical jargon.

- Each use case shows the steps that the actor follows to perform an operation.

- The use cases comply with the graphic notation defined in UML modeling language.

CONFIGURATION MANAGEMENT PLAN REVIEW

CONFIGURATION MANAGEMENT PLAN REVIEW

Clayton Rooke, as Quality Manager, must carry out the revision of the Configuration Management Plan, to do so he must verify that it complies with the following:

- The project includes a Configuration Management Plan for the control and management of changes in which the activities to be carried out are established that allow the control and management of changes in the project.

- The Configuration Management Plan complies with IEEE Std. 828 - 2005: "IEEE *Standard for Software Configuration Management Plans*" and ANSI/IEEE Std. 1042 - 1987: "*IEEE Guide to Software Configuration Management*".

- The management of the configuration defined in the SCM is carried out during all phases of the software project development, including maintenance and change control.

- The SCM describes a change and version control mechanism that ensures the production of quality software.

- The MTS includes the procedure for generating the necessary documentation for recording and monitoring the changes that occur during the development of the project.

REVIEW OF PROJECT ESTIMATION AND PLANNING

REVISION OF ESTIMATE

When planning a project, an estimate of the cost and human effort required must be obtained. Estimation is one of the crucial activities in the software project management process, necessary for project planning.

Clayton Rooke, as Quality Responsible, must make the revision of the estimate made for the software development project, for this he must review the following:

- The method used to estimate the effort for the development of the software project uses size-oriented metrics based on points of use cases.

- Before each iteration, verify that the estimate has been made taking into account the use cases included in the estimate.

- The use case points for each of the iterations have been calculated following the procedure established for this estimation method which includes the following steps:

    - Classify each iteration between actor and chaos of use according to its complexity and assign a weight according to it.

    - Calculate the complexity of each use case according to the number of transactions or steps in the case.

    - Calculate the Unadjusted Use Case Points of the iteration.

    - Calculate technical and environmental complexity factors.

    - Calculate Adjusted Use Case Points.

- Once the use case points have been obtained for an iteration, verify that the corresponding effort required to carry them out in that iteration has been calculated from them.

PLANNING REVIEW

Planning is the process of establishing objectives and choosing the means to achieve them. It is essential to carry out an analysis of the project in order to foresee from the beginning and during the development of the project the situations that may arise and to create the necessary conditions to be able to solve them or minimize the consequences that they may have on the development of the project and the achievement of the objectives.
Clayton Rooke, as Quality Manager, must carry out the revision of the planning made for the software development project, for this he must verify the following:

- A prioritisation of use cases to be developed has been carried out and the iterations that will make up the complete development of the software and the use cases included in each of them have been defined.

- An estimation of each iteration has been made based on Use Cases. Based on this estimate, planning will be carried out.

- Before starting an iteration, a planning of the iteration will be done based on the estimation of the effort needed according to the points of use cases.

- The planned planning for the development of the software project will be adapted and updated as the project progresses.

- Planning includes how many people should participate in the project team, what technical skills are needed, when to increase the number of people and who will participate.

- The planning done defines how the team that will work on the software development project will be organized.

- The planning follows the methodology applied to the software development project which is, in this case, incremental iterative based on use cases.

- A Gantt chart is included, representing all the activities to be carried out throughout the project development period. The diagram connects the different activities based on their relationships of precedence and defines the estimated resources and times for each activity.

- The Gantt chart reflects the tasks and key dates, the milestones and the dependency between tasks.

- The quality metrics to be applied to the planning carried out will be

    - Speed at which objectives or requirements are completed in each iteration

- Urgency and priority of the completed requirements, to check if there is any misalignment with the project objectives and the organization's strategy.
- Requirements completed in iteration.
- Built-in changes and added requirements on the initial scope of iteration
- Number of requirements completed out of total requirements.
- Deviation of project results from initial planning
- Budget available, budget spent and financial deviation from initial planning.
- Customer satisfaction with regard to the results obtained.

TEST PLAN REVIEW

TEST PLAN REVIEW

Clayton Rooke, as Quality Manager, must carry out the revision of the Test Plan, for this he must do the following:

- It should be checked that there are rules for carrying out the tests so that it is possible to verify that these tests have been carried out, as well as indicating how to act in the event of differences between the expected result and the result obtained.

- A traceability matrix must be carried out to ensure that there is evidence to verify all software requirements.

REVIEW OF THE PRODUCTS OF THE ANALYSIS PROCESS

REVIEW OF USE CASES IN EXPANDED FORMAT

Clayton Rooke, as Quality Responsible, must carry out the revision of the Use Cases in expanded format, for this he must do the following:

- From each high-level use case, an expanded use case has been built, in each iteration.
- Each expanded use case is composed of two sections, the header that includes the name, actors, description and type of use case, and the body that describes typical events and alternatives to typical events.
- Expanded use cases define the initiator of the use case.
- The body of the use case consists of two columns describing the actions of the actor and the system responses to them.

REVIEW OF THE CONCEPTUAL MODEL OF THE ANALYSIS

Clayton Rooke, as Quality Manager, must carry out the revision of the Conceptual Model, for this purpose the following must be verified:

- The analysis model represents the aspects of the problem in a way that is close to the concepts of the problem domain and describes the main characteristics of the system. The analysis model carried out in each of the iterations that make up the project will be validated.
- The conceptual model does not include implementation decisions. It will also be verified that it is independent of the implementation.
- The conceptual model complies with the graphic notation of the UML modeling language. You should also check that the notation has the necessary level of detail to represent the problem, without being overloaded.
- The conceptual model has been made through an object model or class diagram (without methods) that defines the system properties. The entities and the relationships between them have been identified for each iteration.
- The quality metrics to be applied to the conceptual model resulting from the analysis in each iteration are the following:
    - Semantic quality: correspondence between the model and the domain, i.e. the model reflects the domain. The validity of the model will be verified,

i.e. that all the facts included in the model are correct and relevant to the domain.

- Completeness: the model will be checked to ensure that all facts are correct and relevant to the domain.

- Language quality: the modeling language used to capture the domain is a language that is easy to understand by all participants. The formalization of the language allows the execution of the system.

- Syntactic quality: there is a correspondence between the externalization of the model and the extension of the language in which the model is written.

REVIEW OF OPERATING CONTRACTS

Clayton Rooke, as Quality Manager, must carry out the revision of the operation contracts that are generated, for this purpose the following must be verified:

- For each case of use, there must be a contract of operation for each action of the actor.

- Each operating contract will consist of the following fields: name, responsibilities, cross references, notes, exceptions, output, pre-conditions and post-conditions.

- Cross-references in the contract shall correspond to references to the requirements defined in the project that are resolved with the use case to which the operation contract belongs.

REVIEW OF THE DESIGN PROCESS PRODUCTS

CLASS DIAGRAM REVIEW

Assessing whether the design obtained meets the required quality level is important in order to know the effectiveness of the processes that have been modeled and whether or not they require great effort for their implementation.

Evaluating design class models by applying metrics allows for the detection of shortcomings and potential improvements from early stages of product development, preventing them from spreading to subsequent phases and enabling the creation of a robust system from its conception.

Clayton Rooke, as Quality Responsible, will have to carry out the revision of the Class Diagrams, for this he will have to check the following:

- Class diagrams will be made for each iteration with UML and the design will be totally independent of the implementation.

- The comprehensibility of the model or facility with which the class diagram can be understood, the analyzability of the model or facility offered by the class diagram to discover its deficiencies or errors, and the modifiability of the diagram or facility offered by the diagram to make a specified modification, either by error, by a concept not taken into account or by a change in requirements, shall be measured.

- The following metrics will be used to measure the structural complexity of the class diagrams:

    - Number of classes: total number of classes.

    - Number of attributes: total number of attributes.

    - Number of methods: total number of methods.

    - Number of partnerships: total number of partnerships.

    - Number of aggregations: total number of aggregation ratios.

    - Number of dependencies: total number of dependency relationships.

    - Number of generalizations: total number of generalization ratios.

    - Number of generalization hierarchies: total number of generalization hierarchies

    - Number of aggregations: total number of aggregation ratios.

- WMC: class weighted methods, according to their complexity.

- Maximum ITL: is the maximum ITL value obtained for each class in a class diagram. For a class within a generalization hierarchy, it is the length of the longest path from the class to the root of the hierarchy.

- Maximum HAgg: is the maximum HAgg value obtained for each class in the class diagram. For a class within an aggregation hierarchy it is the length of the longest path from the class to the leaves.

- The proposed metrics are highly related both to maintenance time and to the comprehensibility, analyzability and modifiability of the designed class diagram.

REVIEW OF SEQUENCE DIAGRAMS

Clayton Rooke, as Quality Manager, must carry out the revision of the sequence diagrams generated in the project during the design phase of each iteration, for this purpose the following must be verified:

- For each use case, sequence diagrams have been designed that define both the typical course and the atypical courses of the events defined in them.

- The sequence diagrams show the interaction represented by the sequence of messages between the class instances and actors. The diagrams show instances and events that describe the interaction between the classes.

- Time flows down the diagrams and shows the control flow from one participant to another.

- The UML notation is followed in the definition of the diagrams. The elements included in the sequence diagram are:

    - Name of the sequence diagram.

    - Lifelines for actors and class instances.

    - Messages between instances that define the method that the message calls on the receiving lifeline. In addition, the receiving line is linked to an interface or class.

    - Loops indicate the number of times the loop is executed if known.

REVIEW OF STATE DIAGRAMS

Clayton Rooke, as Quality Manager, must carry out the revision of the state diagrams generated in the project during the design phase of each iteration, for this purpose the following must be verified:

- The defined state diagrams describe the behavior of the system, with each diagram showing the behavior of a single object during its entire life cycle.

- State diagrams contain states and transitions, and the transitions between them include the corresponding events or actions.

- The state diagram shows all possible states that the object goes through during its life in the application as a result of the events that reach it.

- There is an initial state and a final state and all states represented in the diagram are accessible.

6.Estimation

Table 20: Use Case Point Estimation Method

Adjusted Use Case Points (UCP) = UUCP * TCF * EF		207.30	
Person Hours Mulitiplier (PHM) (Per use case)	* A value of 0 means too risky to proceed	20.00	hours.use-case
Effort in Person Hours = UCP * PHM (just coding)		4146.07	hours.man
Effort in Person Hours whole project		10365.16	hours.man
Esfuerzo Meses Persona Estimados en el Proyecto		89.35	MM
Time estimated using COCOMO II Organic Mode	$Tdev=2.5(MM)^{0.38}$	13.78	Months
Average Team Size (Full Time)	Team Size =MM/Tdev	6.48	People
Cost		92124.85	Euros
Hours worked per month	116		
Average Monthly Salary (euros)	1031		

Table 21: Phase Percentage Breakdown

Phase	Percentage
Analysis	10.00%
Design	20.00%
Coding	40.00%
Test	15.00%
Extra (other activities)	15.00%

Table 22: Phase Hours Breakdown

Phase	Hours.man	M.M	Months
Analysis	1036.52	8.94	5.7
Design	2073.03	17.87	7.5
Coding	**4146.07**	35.74	9.7
Test	1554.77	13.40	6.7
Extra (other activities)	1554.77	13.40	6.7
Total	*10365.16*	89.35	13.8

7.Planning

Summary of Gantt Chart

Phase 0
Start Date: Mon 21/02/22
End Date: Fri 18/03/22
Resources: Carried out solely by the Project Manager. There is no overload.

Planning and Requirement Specification
Start Date: Fri 18/03/22
End Date: Mon 04/04/22
Resources: Carried out by the Project Manager and Software Engineers 1 and 2. There is no overload.

Construction Phase Iteration 1
Start Date: Mon 04/04/22
End Date: Tue 07/06/22
Resources: Carried out using all resources. There is no overload

Construction Phase Iteration 2
Start Date: Tue 07/06/22
End Date: Wed 10/08/22
Resources: Carried out using all resources. There is no overload

Construction Phase Iteration 3
Start Date: Wed 10/08/22
End Date: Thu 13/10/22
Resources: Carried out using all resources. There is no overload

8.Planning and requirements specification

8.1 Feasibility study

IDENTIFYING THE SCOPE OF THE SYSTEM

SiestAI will develop the application, Break Buddies, to be used in TD offices around the world. Break Buddies is a web platform that integrates with corporate calendars to schedule tailored social and individual activities amongst large groups, specifically within large-sized organizations with in-person function. Our product proposal encourages taking breaks while connecting with others by automating the decisions and removing any barriers. It makes it simple for people to take breaks during the workday by automatically organizing them together in convenient time slots. Based on each participant's interests and preferences, it provides content in the form of video or audio to facilitate the activity. The mission is to encourage work-life balance and social interaction in the office by collectively simplifying break scheduling. This will help boost employee productivity and morale, as well as promote healthier habits to reduce the rate of burnout.

IDENTIFICATION OF STAKEHOLDERS IN THE SYSTEM

BreakBuddies is made for companies with a large number of in-person office employees and those who value sustainable work habits for their employees. BreakBuddies' customer is TD Bank, which is the largest commercial bank in Canada, offering a range of financial services and products to 15 million customers, with 90,000 employees worldwide. BreakBuddies' users are the employees at TD.

8.1.1 Requirements definition

The requirements are going to be described as follows:

Identifier:	
Name:	
Priority:	**Source:**
Necessity:	
Clarity:	**Verifiability:**
Stability:	
Description:	

- The identification of the requirements will be done in the following way:
    - Identifier: UX-nnn, where
    - U: indicates that this is a user requirement
    - X: Placeholder for requirements of the following types
        - F: Functional
        - I: Interfaces
        - P: Performance
        - S: Security
        - R: Reliability
        - A: Availability
        - M: Maintainability
        - T: Portability
    - nnn: Consecutive numbers to identify a requirement
- The name field summarizes the requirement

- The priority will have one of the following values:
  - High
  - Medium
  - Low
- The source field can have one of the following values:
  - Customer
  - Analysts
- The necessity field will have one of the following values:
  - High
  - Medium
  - Low
- The clarity field will be assigned one of the following values:
  - High
  - Medium
  - Low
- The verifiability field can have one of the following values:
  - High
  - Medium
  - Low
- Stability describes the duration of the requirement over the life of the software.
  - High
  - Medium
  - Low
- The description field serves to explain the requirement.

Functional requirements

Identifier: UF-001		
Name:	System should recommend one activity for each user every day	
Priority: High		**Source:** Customer
Necessity: High		
Clarity: High		**Verifiability:** High
Stability: High		
Description:	Each day, the system must recommend a break activity for each user from the available options.	

Identifier: UF-002		
Name:	Recommended activity should match at least one of the user's interests	
Priority: High		**Source:** Customer
Necessity: High		
Clarity: Low		**Verifiability:** Low
Stability: High		

Description:	Using the user's chosen interests, the system should select for them an activity from the available options which is related to at least one of their interests, so that the user is likely to be interested in the chosen activity.

Identifier: UF-003	
Name:	The number of people attending an activity should fall within a specified range
Priority: High	**Source:** Customer
Necessity: High	
Clarity: Medium	**Verifiability:** High
Stability: High	
Description:	Each of the available activities (sourced from online or from administrators) will specify a range of attendees which is acceptable for that activity. For example, some individual activities may be done independently, while some group activities may require at least N people to function properly. The system should schedule activities such that number of people attending the activity, meaning the number of users who add the activity to their calendar, falls within the specified range.

Identifier: UF-004	
Name:	Recommended break activity must fit in the user's calendar
Priority: Medium	**Source:** Customer
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	The system must recommend a time for the activity which does not conflict with the user's calendar, meaning that the activity time does not overlap with the time of any event in the user's calendar.

Identifier: UF-005	
Name:	Users should be able to provide feedback for each activity
Priority: Low	**Source:** Customer
Necessity: Medium	
Clarity: Medium	**Verifiability:** Medium
Stability: High	
Description:	After completing a break activity, each user should be able to provide feedback on the activity using a feedback form.

Identifier: UF-006	
Name:	Administrators should be able to modify the set of available activities
Priority: Low	**Source:** Customer
Necessity: Medium	

Clarity: Medium	**Verifiability:** High
Stability: High	
Description:	Administrators should have access to the database of available activities that the system chooses from, should be able to create and delete activities, and should be able to modify their parameters.

Identifier: UF-007	
Name:	Users should be able to choose a different activity than the recommended one
Priority: Low	**Source:** Customer
Necessity: Medium	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	If the user does not want to attend the recommended activity, they should be able to see a list of the available activities scheduled for that day and choose to attend a different one

Identifier: UF-008	
Name:	Users should be able to specify their interests
Priority: Medium	**Source:** Customer
Necessity: High	
Clarity: Medium	**Verifiability:** Medium
Stability: High	
Description:	Each user should be able to create and modify a list of their interests, which the system users to ensure that recommended activities relate to their chosen interests.

Identifier: UF-009	
Name:	Users should receive a calendar invite to recommended activities
Priority: High	**Source:** Customer
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	Once an activity has been selected for a user, they should receive an invite to the activity, which can be accepted or rejected. If accepted, the activity should be added to the user's calendar, and the calendar item should contain any required information, such as the room in which the activity will take place, or the link to the online component of the activity.

Non-functional requirements

Performance requirements

Identifier: UP-001

Name:	95% of transactions must be completed in less than 1 second
Priority: High	**Source:** Customer
Necessity: Medium	
Clarity: Medium	**Verifiability:** High
Stability: High	
Description:	We want our app to be responsive, so users are more likely to use it and have a good time using it. Some services the app uses may take time to load, hence the 5 percent buffer for the sub-1 second requirement

Identifier: UP-002	
Name:	99% of transactions must be completed in less than 2 seconds
Priority: High	**Source:** Customer
Necessity: Medium	
Clarity: Medium	**Verifiability:** High
Stability: High	
Description:	Most processes will not take over 2 seconds, even using third party services such as YouTube or external links; and we want to minimize the feelings of unresponsiveness felt by users.

Identifier: UP-003	
Name:	Should support 50,000 transactions per second
Priority: High	**Source:** Analyst
Necessity: High	
Clarity: Medium	**Verifiability:** High
Stability: High	
Description:	Our user base is of a large size, considering the size of the companies that we expect to use out product, and hence we need to build capacity for a large amount of transactions and interactions with the system

Identifier: UP-004	
Name:	Should support 10,000 simultaneous users
Priority: High	**Source:** Customer
Necessity: High	
Clarity: Medium	**Verifiability:** High
Stability: High	
Description:	Our User Base is of large companies; who can have thousands of employees; for our current customer; 10, 000 was gathered from their employment statistics

Security

Identifier: US-001	
Name:	All data transfer should be encrypted
Priority: High	**Source:** Analyst
Necessity: High	
Clarity: High	**Verifiability:** High

Stability: High	
Description:	We must ensure that our users' privacy is respected and that external malicious actors do not have access to it.

Identifier: US-002		
Name:	Critical databases will be location redundant	
Priority: High		**Source:** Analyst
Necessity: High		
Clarity: High		**Verifiability:** High
Stability: High		
Description:	Databases that are essential to the system must be kept in locations which cannot be compromised, and have redundancy built in.	

Identifier: US-003		
Name:	Access permissions for the system information may only be changed by the system's data administrator.	
Priority: High		**Source:** Analyst
Necessity: High		
Clarity: High		**Verifiability:** High
Stability: High		
Description:	This power must be authorized as this kind of access can lead to major issues in the system, as well as with the privacy of our clients.	

Identifier: US-004		
Name:	The database may only communicate with the web server and not directly to the client	
Priority: High		**Source:** Analyst
Necessity: High		
Clarity: High		**Verifiability:** High
Stability: High		
Description:	This is a common software industry standard to ensure the system stays secure and the client does not get accessed directly.	

Identifier: US-005		
Name:	User credentials should be kept in an encrypted form	
Priority: High		**Source:** Analyst
Necessity: High		
Clarity: High		**Verifiability:** High
Stability: High		
Description:	Industry practice to ensure the anonymity and security of users	

Reliability

Identifier: UR-001	
Name:	Total permissible UI related incidents should be less than 1 reported error per thousand uses
Priority: Medium	**Source:** Analyst
Necessity: Low	
Clarity: Low	**Verifiability:** Medium
Stability: Medium	
Description:	We want our program to be relatively bug free to reduce user frustration. UI items are the interaction points for users so we want them to be robust to encourage use. The verifiability of this is lower as UI bugs can often go unreported.

Identifier: UR-002	
Name:	Total permissible content availability issues should affect at most 10% of the content catalog and must be resolved by takedown or fix after less than 5 complaints per organization.
Priority: Medium	**Source:** Analyst
Necessity: Medium	
Clarity: High	**Verifiability:** High
Stability: Low	
Description:	Ensuring content is available and usable is essential to the platform. As they come from third parties in many cases, there is a high chance that content availability issues will arise from region issues, or disappear due to takedowns on the source platform. We still want to avoid this so we have systems in place for checking content availability.

Availability

Identifier: UA-001	
Name:	The system should have 99.99% availability
Priority: High	**Source:** Analyst
Necessity: Medium	
Clarity: Medium	**Verifiability:** High
Stability: High	

Description:

Duration	Acceptable downtime
Downtime per year	52min 35.7s
Downtime per month	4m 23s
Downtime per week	1m 5s
Downtime per day	8.6s

Portability

Identifier: UT-001

Name:	Should work on web browsers of Google Chrome, Safari, Firefox and Microsoft Edge	
Priority: High		**Source:** Analyst
Necessity: High		
Clarity: Medium		**Verifiability:** High
Stability: High		
Description:	Ensuring that product is used on all major browsers which our clients may be using is essential for increasing the ubiquity and reducing pain in the product.	

Identifier: UT-002		
Name:	Responsiveness – should work on a variety of mobile/desktop screen sizes	
Priority: High		**Source:** Analyst
Necessity: High		
Clarity: High		**Verifiability:** High
Stability: High		
Description:	Ensures product is accessible to all users, and hence increases the likelihood that users will utilize the service available to them.	

Identifier: UT-003		
Name:	System should support Mac OS, Windows 7, Windows 10/11, Android, iOS	
Priority: High		**Source:** Analyst
Necessity: High		
Clarity: Medium		**Verifiability:** Medium
Stability: High		
Description:	Ensuring that product is used on all major systems which our clients may be using is essential for increasing the ubiquity of the product and reducing pain for the users	

Identifier: UT-004		
Name:	Front end will support Html5 and JavaScript	
Priority: High		Source: Analyst
Necessity: High		
Clarity: High		Verifiability: High
Stability:	High	
Description:	Ensures product is usable on all the major industry standard platforms that are used by browsers to display web pages	

Common interface requirements

User Interfaces

Identifier: UI-001		
Name:	App colors must be green to match TD Branding	
Priority: High		**Source:** Customer

Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	The main background color for all windows in the application will be green, matching the client branding, specifically having a hexadecimal RGB color value of 0x00A221.

Identifier: UI-002	
Name:	Interface elements such as fonts and images must be responsive for mobile and desktop viewing
Priority: High	**Source:** Customer
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	Interface elements such as fonts and images must be responsive for mobile and desktop viewing as the app will be integrated into calendar apps that have this existing functionality. Responsive web design will allow for easy reading on various devices and benefit users with disabilities.

Identifier: UI-003	
Name:	Color contrast of text and interactive elements must be sufficient
Priority: High	**Source:** Customer
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	Text and interactive elements must have color contrast ratio of at least 4:5:1, which is established by the Web Content Accessibility Guidelines (WCAG) 2.0. This is to ensure that any web content Is accessible to a wider range of people, including ones with disabilities.

Software interfaces

Identifier: UI-004	
Name:	System should integrate with third party Calendar apps with existing scheduling functionality (Google, Outlook)
Priority: High	**Source:** Customer
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	Integrate app with third party calendar apps that have existing scheduling functionality such as Google Calendar and Microsoft Outlook. This is to enable the platform to have access to existing events and schedule new events into users' calendar.

Identifier: UI-005

Name:	System data should be stored in a database
Priority: High	**Source:** Analyst
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	Our database must store the system's data such as user information, event, and activity information.

Identifier: UI-006	
Name:	System should be hosted on AWS
Priority: High	**Source:** Analyst
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	AWS is a cloud computing platform which will be used to host our app and make it available for our users.

Identifier: UI-007	
Name:	System must integrate with workplace/enterprise software systems
Priority: High	**Source:** Customer
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	Integrate app with enterprise software systems for secure authentication and login with corporate accounts.

Communication interfaces

Identifier: UI-008	
Name:	System must communicate over the web using HTTPS
Priority: High	**Source:** Analyst
Necessity: High	
Clarity: High	**Verifiability:** High
Stability:	High
Description:	Web applications use HTTPS to communicate over the internet. Our app will need to have a client and server which will communicate over HTTPS.

Identifier: UI-009	
Name:	Server and database should communicate using the CRUD paradigm
Priority: High	**Source:** Analyst
Necessity: High	
Clarity: High	**Verifiability: High**
Stability: High	
Description:	The server will retrieve and store information for the app in a database using create, read, update, and delete operations.

Identifier: UI-010	
Name:	System must communicate with third-party systems & APIs
Priority: High	**Source:** Analyst
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	Web applications use HTTP to communicate over the internet. Our app will communicate with 3rd party applications such as a calendar apps and enterprise software over HTTP. Our company and third parties will have to come to an agreement to use each other's systems, and our company may have to pay a subscription fee to use third-party APIs.

Other requirements

Identifier: UO-001	
Name:	Ensure our privacy policies are catered to the unique legal requirements of different countries; and multiple versions of region-specific Terms of Service are provided.
Priority: High	**Source:** Analyst
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	Ensures our product meets the legal requirements of all countries it is used in and avoids legal trouble for us.

Identifier: UO-002	
Name:	Application and data handling procedures will be fully compliant to GDPR (General Data Protection Regulation)
Priority: High	**Source:** Analyst
Necessity: High	
Clarity: High	**Verifiability:** High
Stability: High	
Description:	Ensures our product fits the most standard data protection system used nearly worldwide

Identifier: UO-003	
Name:	Ensure that the content on the platform is moderated to fit the legal requirements of all countries it will be used in
Priority: High	**Source:** Analyst
Necessity: High	
Clarity: Medium	**Verifiability:** Medium
Stability: High	

Description:	Ensures the content does not violate any legal requirements of the countries and regions it is operating in, and hence protects us from legal trouble.

8.1.2 Study of alternative solutions

Alternative solutions are demonstrated, which can satisfy all the requirements of the function and non-functional requirements.

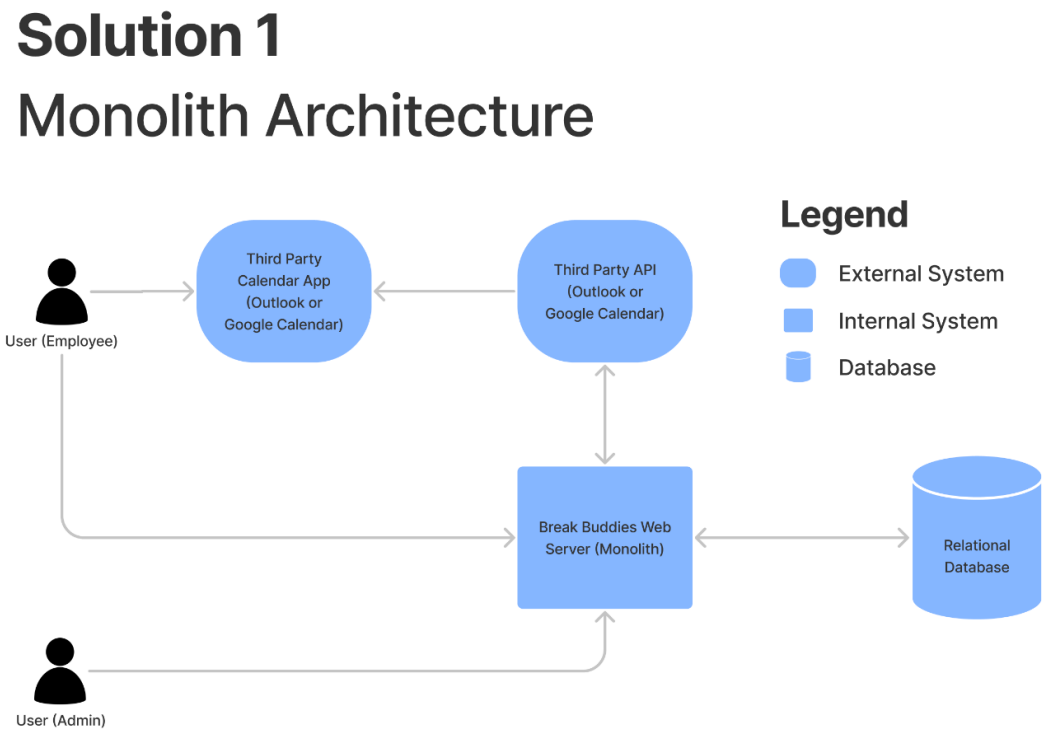


Figure 4: Monolith Architecture (Solution 1 diagram)

In Solution 1, a monolithic architecture is used. A monolith constitutes of a single server that acts as the handles and responds to requests from the clients (employees and admins) and performs the business logic of the application. The main advantage of this approach is simplicity. The drawbacks are that the scalability of the application can be difficult to manage (lowering performance), and the development of the application by multiple developers can be complex to manage.

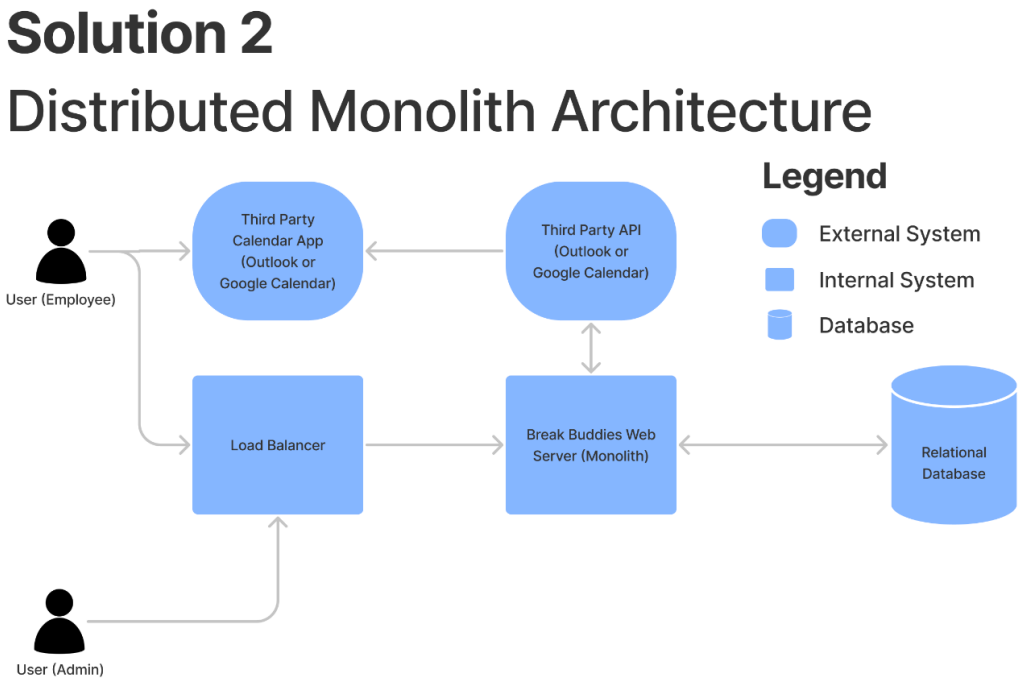


Figure 5: Distributed Monolith Architecture (Solution 2 diagram)

In Solution 2, a distributed monolithic architecture is used. Like the monolithic architecture in Solution 1, a single monolith server is used; but the difference is a load balancer communicated to the clients and the monolith server is distributed across different duplicated webservers across different geographies. Although much more complex than Solution 1, duplicating the servers allows the application to balance the load of users across multiple servers increasing the performance and scalability of the application. The downside is that the complexity to develop, maintain and test is increased compared to Solution 1.

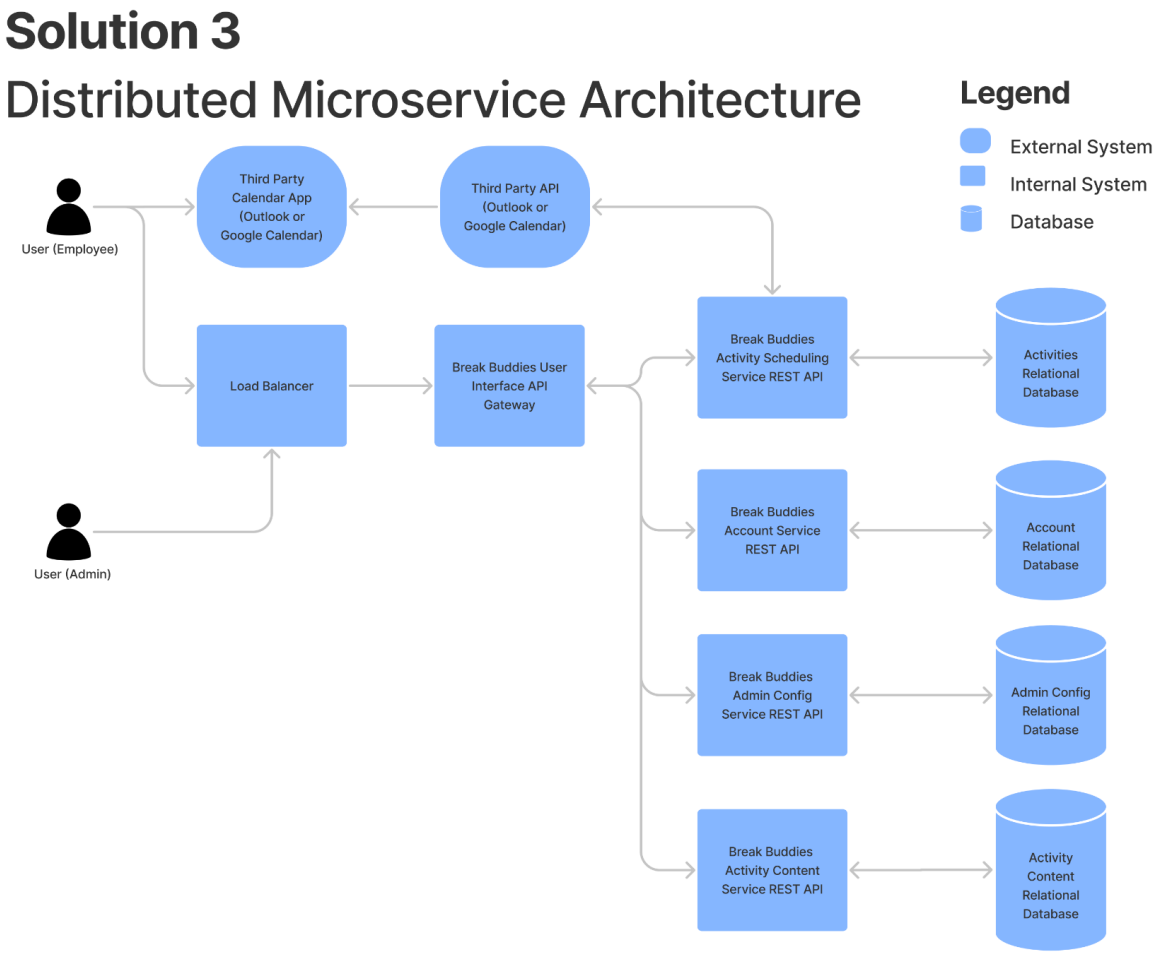


Figure 6: Distributed Microservice Architecture (Solution 3 diagram)

In Solution 3 a distributed microservice architecture is used. In a microservice architecture, instead of a monolith acting as the sole server, multiple independent components are used that communicate to each other to accomplish the business logic requirements. The upsides of this approach are lowered impact of failures (decreased downtime), increased scalability, increased performance, and easier extendibility. The downsides are that there are increased development costs.

8.1.3 Valuation of alternatives

In Table 23, a weighted decision matrix is used to evaluate each of the solutions (scored on a scale from 0 to 7)

Table 23: Weighted Decision Matrix

	Functionality	Interface	Performance	Security	Complexity	Development Time	Total
Solution 1 Monolith	7	7	3	5	5	7	225
Solution 2	7	7	5	5	4	5	223

Distributed Monolith							
Solution 3 Distributed Microservice	7	7	7	7	3	3	205
Weight	5	5	5	7	7	10	

8.1.4 Solution selection

Solution 1, the monolith architecture is the best solution as it has scored the highest in our decision matrix. The weighting in the matrix was determined by the perceived importance of each product feature. Following the assumption that each solution meets all identified requirements, development time was prioritised to reduce costs. A lower complexity and security were weighted moderate importance because higher complexity would result in decreased maintainability. The functionality and interface would be all satisfied by each solution as the functionality of the architectures would be in parity. Performance was weighted low because the scale of users would be relatively low when starting out, making it much easier to meet the performance requirements.

8.2 Use case model and traceability matrix

The use case model is shown in Figure 7.

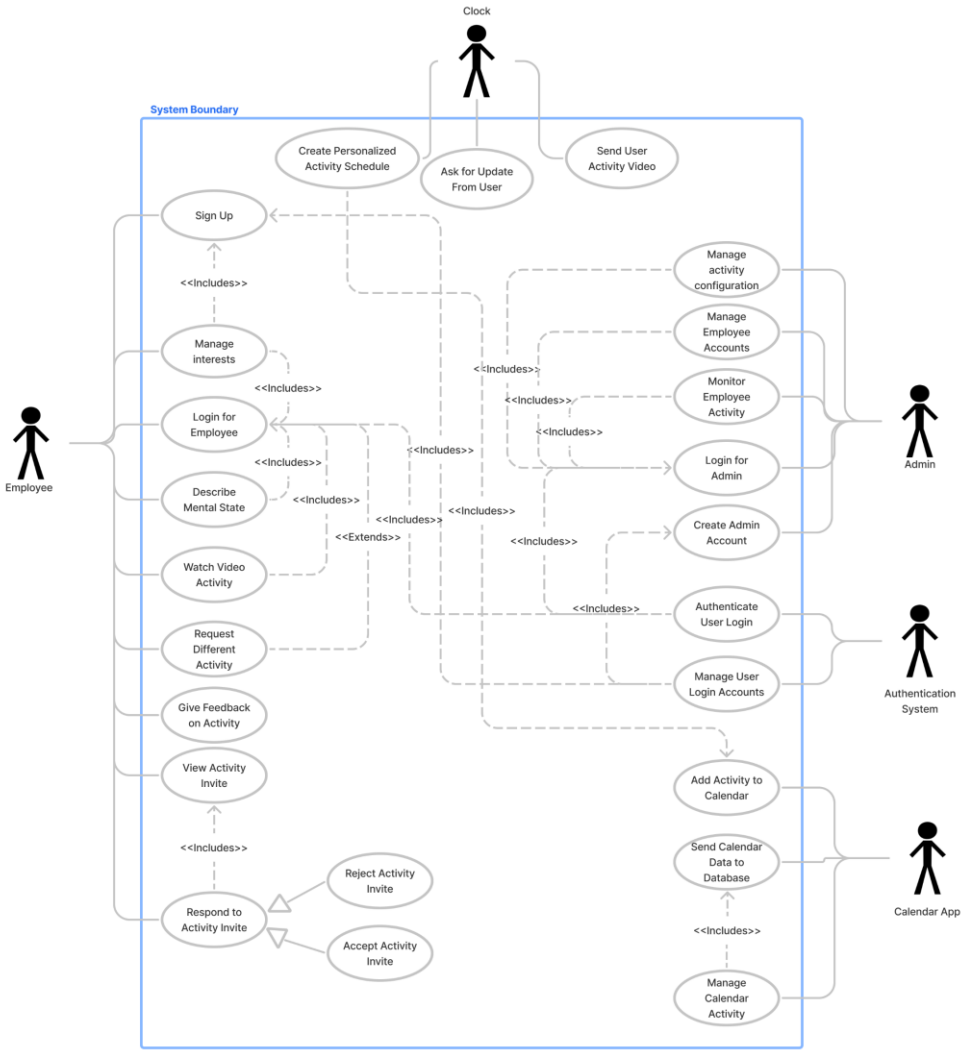


Figure 7: Use case model for Break Buddies

The traceability matrix is shown in Table 24. In general, use cases satisfy one or more functional requirements. Many use cases also satisfy the same non-functional requirements. For example, some requirements related to performance of the system will be fulfilled by implementing each use case in a performant way.

Table 24: Traceability matrix. The use cases are split over multiple sections to fit.

	Use Cases						
Requirements	Sign Up	Manage interests	Login for Employee	Describe Mental State	Watch Video for Activity	Request Different Activity	Give Feedback on Activity
UF-001							
UF-002							
UF-003							
UF-004							
UF-005							X

UF-006							
UF-007						X	
UF-008		X					
UF-009							
UP-001	X	X	X	X	X	X	X
UP-002	X	X	X	X	X	X	X
UP-003	X	X	X	X	X	X	X
UP-004	X	X	X	X	X	X	X
US-001	X	X	X	X	X	X	X
US-002	X	X		X			X
US-003							
US-004	X	X	X	X			X
US-005	X		X				
UR-001	X	X	X	X	X	X	X
UR-002	X	X	X	X	X	X	X
UA-001	X	X	X	X	X	X	X
UT-001	X	X	X	X	X	X	X
UT-002	X	X	X	X	X	X	X
UT-003	X	X	X	X	X	X	X
UT-004	X	X	X	X	X	X	X
UI-001	X	X	X	X	X	X	X
UI-002	X	X	X	X	X	X	X
UI-003	X	X	X	X	X	X	X
UI-004							
UI-005	X	X		X			X
UI-006	X	X		X			X
UI-007	X		X				
UI-008	X	X	X	X	X	X	X
UI-009		X					
UI-010					X		
UO-001	X						
UO-002	X			X			X
UO-003		X			X		

	Use Cases						
Requirements	View Activity Invite	Respond to Activity Invite	Send User Activity Video	Ask for Update from User	Create Personalized Activity Schedule	Manage activity configuration	Manage employee accounts
UF-001					X		
UF-002					X		
UF-003					X		
UF-004					X		
UF-005							
UF-006						X	
UF-007							
UF-008							
UF-009							
UP-001	X	X	X	X		X	X
UP-002	X	X	X	X		X	X

UP-003	X	X	X	X		X	X
UP-004	X	X	X	X		X	X
US-001	X	X	X	X		X	X
US-002					X	X	X
US-003							X
US-004	X	X	X	X	X	X	X
US-005							
UR-001	X	X				X	X
UR-002	X	X	X	X	X	X	X
UA-001	X	X	X	X	X	X	X
UT-001	X	X				X	X
UT-002	X	X				X	X
UT-003	X	X				X	X
UT-004	X	X				X	X
UI-001	X	X				X	X
UI-002	X	X				X	X
UI-003	X	X				X	X
UI-004	X	X					
UI-005						X	X
UI-006	X	X	X	X	X	X	X
UI-007							
UI-008	X	X	X	X		X	X
UI-009						X	X
UI-010							
UO-001	X	X	X	X	X	X	X
UO-002	X	X	X	X	X	X	X
UO-003			X			X	

	Use Cases							
Requirements	Monitor Employee Activity	Login for Admin	Create Admin Account	Add Activity to Calendar	Send Calendar Data to Database	Manage Calendar Activity	Authenticate User Login	Manage User Login Account
UF-001								
UF-002								
UF-003								
UF-004					X			
UF-005								
UF-006								
UF-007								
UF-008								
UF-009				X		X		
UP-001	X	X	X			X	X	X
UP-002	X	X	X			X	X	X
UP-003	X	X	X			X	X	X
UP-004	X	X	X			X	X	X
US-001	X	X	X	X	X	X	X	X
US-002			X		X			
US-003								
US-004	X	X	X	X	X	X	X	X
US-005		X					X	X

UR-001	X	X	X					X
UR-002	X	X	X			X		X
UA-001	X	X	X	X	X	X	X	X
UT-001	X	X	X					X
UT-002	X	X	X					X
UT-003	X	X	X					X
UT-004	X	X	X					X
UI-001	X	X	X					X
UI-002	X	X	X					X
UI-003	X	X	X					X
UI-004				X	X	X		
UI-005	X	X	X	X	X	X	X	X
UI-006	X	X	X	X	X	X	X	X
UI-007		X						
UI-008	X	X	X	X	X	X	X	X
UI-009					X	X		
UI-010		X	X	X	X	X	X	X
UO-001			X					
UO-002	X	X	X		X		X	X
UO-003								

8.3 Use cases high level description

The use case descriptions use the following format.

Use Case: Use case name

Actors: List of actors (external agents), indicating who initiates the use case. Actors are usually roles that a human being plays, but it can be any kind of system

Type:
1. primary, secondary or optional
2. essential or actual (real)

Description: Typical course of events (ex. A customer arrives at the ATM, inserts his card, identifies himself and requests a withdrawal operation for a specific amount. The cashier gives you the requested money after checking that the operation can be carried out. The client takes the money and the card and leaves.)

The use cases are described below.

Use Case: Sign Up
Actors: Employee, authentication system
Type: Primary
Description: An employee visits the Break Buddies website for the first time. They click the Sign Up button and they enter their existing work email credentials. The authentication system validates their credentials. A new account is now created and linked to their work email. The interface is responsive, error-free, and consistent with

TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Manage interests
Actors: Employee
Type: Primary
Description: An employee visits the "Interests" page through their account portal. They are able to select/deselect interests from a given list of activities displayed on the page. After they are done with their selection, they press the 'Save' button to have their interests updated. The interface is responsive, error-free, and consistent with TD branding. The employee can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Login for Employee
Actors: Employee, authentication system
Type: Primary
Description: An employee visits the Break Buddies website and click login. They enter their existing work email credentials. The authentication system validates their credentials and then they are brought to the home page of their portal. The interface is responsive, error-free, and consistent with TD branding. The employee can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Describe Mental State
Actors: System (initiator), Employee
Type: Primary
Description: Everyday at 10 AM, the system prompts the employee to answer how they are feeling for the day by selecting an emoticon that best represents how they are feeling. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Watch Video for Activity
Actors: Employee
Type: Primary
Description: When it's time for the break, an embedded video will be shown on the home page and activities page. Employees can start playing the video and start following the guided activity. The employee can also pause and scrub the video. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication

happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Request Different Activity
Actors: Employee
Type: Primary
Description: On the homepage of the Break Buddies portal, employees can request activities by clicking the ‘Request an Activity’ button. After clicking, a modal will open where the employee fills out the activity name, description, and selects a category they believe it would fall under. At the bottom, they will click ‘Submit’ in order to send their request. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Give Feedback on Activity
Actors: Employee
Type: Primary
Description: After an employee completes an activity, the system immediately prompts the employee to give feedback by presenting different levels of satisfaction (low, medium, and high) for them to select. The employee can select a level and choose to add any additional comments or choose to skip giving feedback. Employee finishes by clicking submit. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: View Activity Invite
Actors: Employee
Type: Primary
Description: An employee will visit the “Activity Invites” page through their account portal to view a list of new or unanswered invites to activities. New invites are indicated with a colored dot next to the tab name. For each activity in the list, they can view activity details by clicking on the activity. A modal will appear with a more detailed description, number of participants, and time/date. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Respond to Activity Invite
Actors: Employee
Type: Primary
Description: On the page, “Activity Invites”, employees can respond directly on the list or in the modal. They can respond by selecting ‘Yes’, ‘No’, or ‘Maybe’ for each

activity. If ‘Yes’ or ‘Maybe’ is selected, the activity will be added to their calendar. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Send User Activity Video
Actors: Clock
Type: Primary
Description: When it is 30 minutes before the scheduled activity, the activity video gets sent to the Break Buddies website. Before the scheduled activity, the video will be hidden from users. When it is time for the activity to start, there will be a countdown and users will get access to the video. The video will play automatically. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Ask for Update from User
Actors: Clock
Type: Primary
Description: The clock asks for users to respond to the personalized activity schedule daily for whether they can attend their events. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Create Personalized Activity Schedule
Actors: Clock
Type: Primary
Description: The clock sends the employee and new personalized activity schedule daily through email and through the Break Buddies system. The interface is responsive, error-free, and consistent with TD branding. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Manage activity configuration
Actors: Admin
Type: Primary
Description: From the administrator home page of their Break Buddies portal, an admin can view and manage a list of activities through an administrator activity management page. They will be able to create, view, delete, and update details about an activity. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Manage employee accounts
Actors: Admin
Type: Primary
Description: From the administrator home page of their Break Buddies portal, an admin can view and manage a list of employee accounts through an administrator employee account management page. They will be able to create, view, delete, and update details about an employee account. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Monitor Employee Activity
Actors: Admin
Type: Primary
Description: From the administrator home page of their Break Buddies portal, an admin can view details about their employee users through an administrator employee activity page. They can view a summary of all employee activity for a given time frame, and also view individual employee activity. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Login for Admin
Actors: Admin, Authentication System
Type: Primary
Description: An admin visits the Break Buddies website and click login. They enter their existing work email credentials. The authentication system validates their credentials and then they are brought to the administrator home page of their portal. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Create Admin Account
Actors: Admin, Authentication System
Type: Primary
Description: An admin account must be configured with a master password to provide the administrator access. An admin visits the Break Buddies website for the first time. They click the Setup Admin button, and they enter their existing work email credentials and the master password to create an admin account. The authentication system validates their credentials. A new account is now created and linked to their work email. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication

happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Add Activity to Calendar
Actors: Calendar App (Outlook or Google Calendar), Clock System
Type: Primary
Description: Once completing the time-based clock system job to schedule activities for the employees, the system sends a request to create a job using the API (Application Program Interface). The system will fetch the required authentication to allow for edit access of the employee's external accounts with either Outlook or Google Calendar. The response is then validated to ensure the activity was added correctly. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Send Calendar Data to Database
Actors: Calendar App (Outlook or Google Calendar), Web Server
Type: Primary
Description: When a new activity is modified from the calendar systems (Outlook or Google Calendar), a payload is sent to the web server to sync the calendar updates of the external system with the internal web server. The response of the request is used to validate that the activity was successfully sent. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Manage Calendar Activity
Actors: Calendar App (Outlook or Google Calendar), Web Server
Type: Primary
Description: A general use case to cover the CRUD (Create, Update, Read, Delete) operations of the scheduled activities. On any modifications of the activity in Break Buddies a request to the external calendar systems' API (Application Program Interface) with be initiated to ensure the parity between systems. The response of the API request is used to validate the CRUD operation is successful. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Authenticate User Login
Actors: Authentication System
Type: Primary
Description: When a user (employee or admin) logs in their user credentials are input into the authentication system and the authentication system checks to see if the credentials are correct, if they are the system grants access, if not the system does not. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Use Case: Manage User Login Accounts
Actors: Authentication System
Type: Primary
Description: When a CRUD operation is performed from the web server for a user account (employees or admins) a request is sent to the authentication system. After verification of the request and the required security checks, the request for the create, read, update, or delete is carried out by the authentication system and a request is sent back to the system that initiated the request. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

8.4 Use cases prioritization -

For prioritization of the use cases we will use the following criteria:

a. If it has a significant impact on the architectural design
b. Whether it leads to a better understanding of the design
c. If it includes complex or high-risk functions
d. If it represents a process of great importance to the business

Determining the importance of each criterion to the business the following weights were selected, which will be applied to a score from 1 to 10 for each use case.

A = 0.3
B = 0.1
C = 0.2
D = 0.4

Table 25: Prioritization Table

Use Case	A 0.3	B 0.1	C 0.2	D 0.4	Sum	Order
Sign Up	5	4	5	6	5.3	14
Manage Interests	7	7	7	9	7.8	2
Login for Employee	6	3	5	4	4.7	20
Describe Mental State	5	6	4	8	6.1	8
Watch Video for Activity	8	7	8	7	7.5	3
Request Different Activity	8	6	6	8	7.4	4
Give Feedback on Activity	4	6	4	6	5	17
View Activity Invite	4	6	5	5	4.8	18
Respond to Activity Invite (Accept, Reject)	4	6	5	5	4.8	18
Send User Activity Video	7	4	5	7	6.3	6
Ask for Update From User	6	5	4	5	5.1	16
Create Personalized Activity Schedule	8	4	8	10	8.4	1
Manage Activity Configuration	8	3	7	7	6.9	5
Manage Employee Accounts	6	3	6	6	5.7	10
Monitor Employee Activity	6	4	4	6	5.4	12
Login for Admin	4	4	4	3	3.6	22
Create Admin Account	6	4	4	3	4.2	21
Add Activity to Calendar	5	3	5	6	5.2	15
Send Calendar Data to Database	7	3	6	6	6	9
Manage Calendar Activity	7	3	5	7	6.2	7
Authenticate User Login	6	4	6	5	5.4	12
Manage User Login Accounts	7	4	6	5	5.7	10

The scores were selected by consulting with members of the team who were most familiar with each use case and using their inputs to form a ranking.

Use Cases were divided into cycles based on their priority in the table, however many have dependencies on other use cases, so cycles aim to create the highest ranked use cases first along with their dependencies.

The first cycle results in the creation of personalized activity schedules, as it is the highest rank, and provides us with a good proof of concept.

The second cycle revolves around implementing the video feature of the project, as it is the next ranked item. It also allows us to show a high detail demonstration of the product.

The third cycle revolves around managing the activities to allow them to be changed and modified.

The fourth cycle involves creating the structures for authentication and user management. These were clubbed in together as it makes sense for the engineering team to build out these use cases simultaneously; and at this point the whole project is near-completion.

The fifth and final cycle is about creating the feedback systems for users and the methods by which the administrators can monitor the usability of the system.

First Cycle

- Manage Interests
- Manage Calendar Activities
- Add Activity to Calendar
- Send Calendar Data to Database
- Create Personalized Activity Schedule

Second Cycle

- Send User Activity Video
- View Activity Invite
- Respond to Activity Invite (Accept, Reject)
- Watch Video For Activity

Third Cycle

- Request Different Activity
- Describe Mental State
- Manage Activity Configuration

Fourth Cycle

- Manage User Login Accounts
- Authenticate User Login
- Sign Up
- Login for Employee
- Login for Admin
- Create Admin Account
- Manage Employee Accounts

Fifth Cycle

- Give Feedback on Activity
- Ask for Update From User
- Monitor Employee Activity

With the prioritization and further breakdown of use cases into development cycles; we can see the more user facing parts of the project are built first; which allows us to see for ourselves how the product will look, and these are all also high-risk activities, so it allows us to make changes if we see issues in them early-on. The back-end engineering aspects come later and are much more difficult to change; they rely on the previous use cases to be completed; and as those are more changeable and flexible it works well that those are completed first.

9.Construction

9.1First Iteration

First Cycle

9.1.1 First iteration analysis

Expanded format use cases description

Use Case: Manage interests
Actors: Employee
Purpose: To let employees input their interests in activities
Overview: An employee visits the “Interests” page through their account portal. They are able to select/deselect interests from a given list of activities displayed on the page. After they are done with their selection, they press the ‘Save’ button to have their interests updated. The interface is responsive, error-free, and consistent with TD branding. The employee can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.
Type: Primary
References: UF-008
Typical Course of Events:

Actor	System
1. Employee opens “Interests” page through their profile page	2. Fetches the employees interested activities and displays it on the UI
3. The employee makes edits to their preferred activities	4. The system updates the user’s new interests

Alternative Courses:

Use Case: Manage Calendar Activities
Actors: Calendar App (Outlook or Google Calendar), Web Server
Overview: A general use case to cover the CRUD (Create, Update, Read, Delete) operations of the scheduled activities. On any modifications of the activity in Break Buddies a request to the external calendar systems’ API (Application Program Interface) with be initiated to ensure the parity between systems. The response of the API request is used to validate the CRUD operation is successful. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.

Type: Primary
References: UF-009
Typical Course of Events:

Actor	System
1. Change is made in the third-party calendar application to the scheduled activity	2. The scheduled activity is updated with the same changes

Alternative Courses:
5. No text is inputted in activity information fields. The save button is disabled.
13. Admin clicks 'Cancel' in confirmation modal. Activity is not deleted.

Use Case: Add Activity to Calendar
Actors: Calendar App (Outlook or Google Calendar), Clock System
Purpose: For system to add activity to employee's third-party calendar
Overview: Once completing the time-based clock system job to schedule activities for the employees, the system sends a request to create a job using the API (Application Program Interface). The system will fetch the required authentication to allow for edit access of the employee's external accounts with either Outlook or Google Calendar. The response is then validated to ensure the activity was added correctly. Communication happens through a private and encrypted web connection to the system server hosted on AWS.
Type: Primary
References: UP-001, UP-002, US-005
Typical Course of Events:

Actor	System
1. At scheduled time tracked by the clock, alert goes out to system to schedule activity	2. System authenticates for edit access to calendar and adds the activity to the third party calendar

Alternative Courses:
2. If there is no response from the external calendar app, report an error.

Use Case: Send Calendar Data to Database
Actors: Calendar App (Outlook or Google Calendar), Web Server
Purpose: To sync calendar updates between the Calendar app and the internal server
Overview: When a new activity is modified from the calendar systems (Outlook or Google Calendar), a payload is sent to the web server to sync the calendar updates of the external system with the internal web server. The response of the request is used to validate that the activity was successfully sent. Communication happens through a private and encrypted web connection to the system server hosted on AWS.
Type: Primary
References:
Typical Course of Events:

Actor	System
1. Change happens in calendar and payload is sent to web server	2. Web server updates the calendar's changes and responds to the request
3.Calendar receives success response	

Alternative Courses:
3. Error response is received, error that update did not sync is displayed to user and system retries.

Use Case: Create Personalized Activity Schedule
Actors: Clock
Overview: The clock sends the employee a new personalized activity schedule daily through email and through the Break Buddies system. The interface is responsive, error-free, and consistent with TD branding. Communication happens through a private and encrypted web connection to the system server hosted on AWS.
Purpose: For the employees to be notified of their enrolled activity
Type: Primary
References: UF-001, UF-009

Typical Course of Events:

Actor	System
1. The clock attempts to schedule an activity	2. The server fetches a list of activities, participants, preferences, calendars and availabilities to ensure participant compatibility

Alternative Courses:
2. An error is reported by the server if there are no compatible parameters

Conceptual Model

For this iteration, the conceptual model is a class diagram. It is shown in Figure 12.

Operation contracts

Format:
Name: Name of the operation and parameters.
Responsibilities: An informal description of the responsibilities that the operation should perform.
Cross-references: functional requirements, use cases, etc.
Notes: Design comments, algorithms, etc.
Exceptions: Exceptional cases. Situations that we should be aware of can happen. It also indicates what is done when the exception occurs.
Output: Outputs that do not correspond to the user interface, such as messages or logs that are sent outside the system. (In most operations of the system this section remains empty)
Pre-conditions: Assumptions about system status before running the operation. Something we don't take into account that can happen when this system operation is called.
Post-conditions: The status of the system after the operation is completed.

Operation Contracts:

Name: scheduleActivities (date: year/month/day)

Responsibilities: Schedule break activities for a given day based on activity options, available rooms, and participant preferences and schedules.
Cross-referencing:

- System functions: UF-001, UF-002, UF-003, UF-004
- Use case: Create Personalized Activity Schedule

Notes: Scheduling algorithm should try to create the best possible schedule such that all users can attend activities they are interested in.
Exceptions: If there are no options for activities, or no available rooms, report an error.
Output:
Pre-conditions: Activity options, available rooms, participant information, participant calendars, and participant activity preferences are saved in the system.
Post-conditions: One or more scheduled activities have been created. Each participant has been assigned one scheduled activity.

Name: addActivity(event: ScheduledActivity)
Responsibilities: Send a calendar invite for a scheduled activity to user calendars.
Cross-referencing:

- System functions: UF-009
- Use case: Add Activity to Calendar

Notes: N/A
Exceptions: If there is no response from the external calendar app, report an error.
Output: A calendar invite sent to an external calendar app.
Pre-conditions: A user has accepted an invite for an activity on the user interface (inviteResponse(true) has been called).
Post-conditions: Each attending user has received a calendar invite for their activity.

Name: showInterests()
Responsibilities: Show the user's preferred activities on the user interface.
Cross-referencing:

- System functions: UF-008
- Use case: Manage Interests

Notes: N/A
Exceptions: N/A
Output: None
Pre-conditions: The user is registered in the system as a Participant and logged in.
Post-conditions: The user can see their preferred activities on the user interface.

Name: updateInterests(Interests: List of PreferredActivities to update)
Responsibilities: Create, change, or delete a user's preferred activities.
Cross-referencing:

- System functions: UF-008
- Use case: Manage Interests

Notes: N/A
Exceptions: N/A
Output: None
Pre-conditions: The user can see their preferred activities on the user interface (showInterests has been called).

Post-conditions: The user has updated or deleted preferred activities, or created new ones.

Name: sendCalendarData(calendarData: user's calendar in external format, participant: Participant whose calendar is being managed)
Responsibilities: Create or update calendar data in the system from an external calendar app.
Cross-referencing:

- System functions: UF-004
- Use case: Send Calendar Data to Database

Notes: N/A
Exceptions: N/A
Output: None
Pre-conditions: Participant is registered in the system, calendar app is connected to the system.
Post-conditions: The user's calendar is created or updated in the system.

Name: updateEvent(CRUD: type of request (create, read, update, delete), event: ScheduledActivity, participants: Participants to update)
Responsibilities: Create, read, update, or delete a ScheduledActivity based on changes made in users' external calendar app.
Cross-referencing:

- System functions: UF-009
- Use case: Manage Calendar Activities

Notes: N/A
Exceptions: N/A
Output: None
Pre-conditions: The user has accepted an activity invite and the activity is in the user's calendar app.
Post-conditions: The scheduled activity has been updated.

9.1.2 First iteration Design

Sequence diagrams

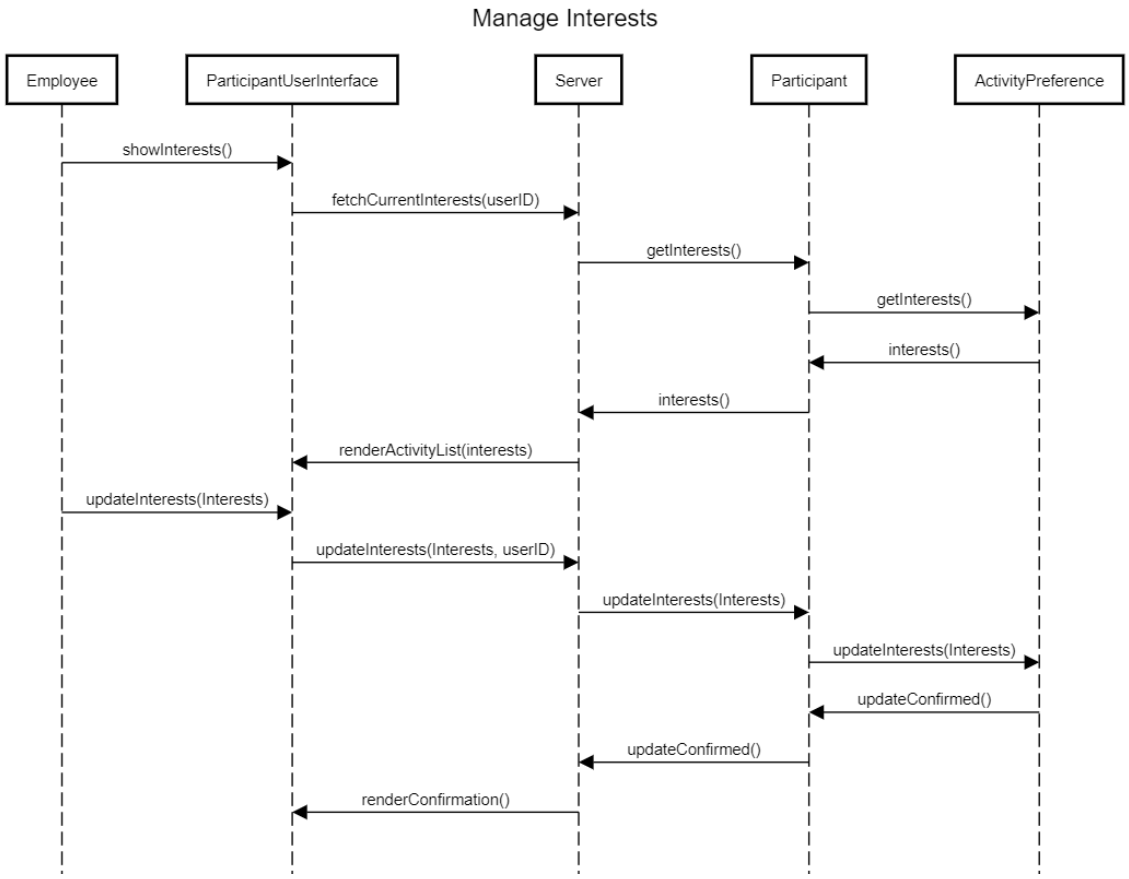


Figure 8: Manage Interests Sequence Diagram

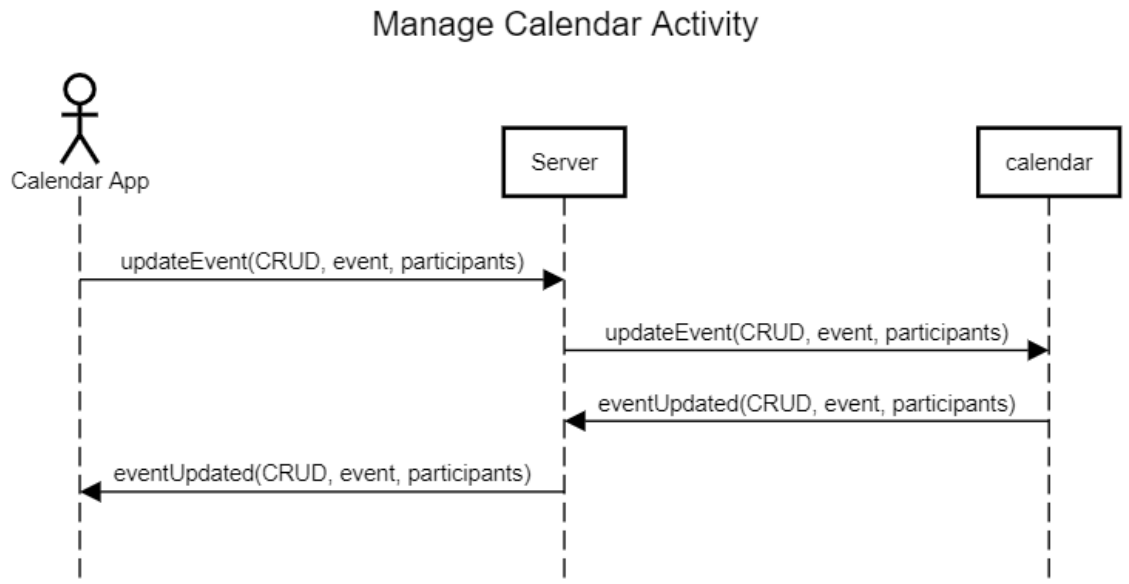


Figure 9: Manage Calendar Activity Sequence Diagram

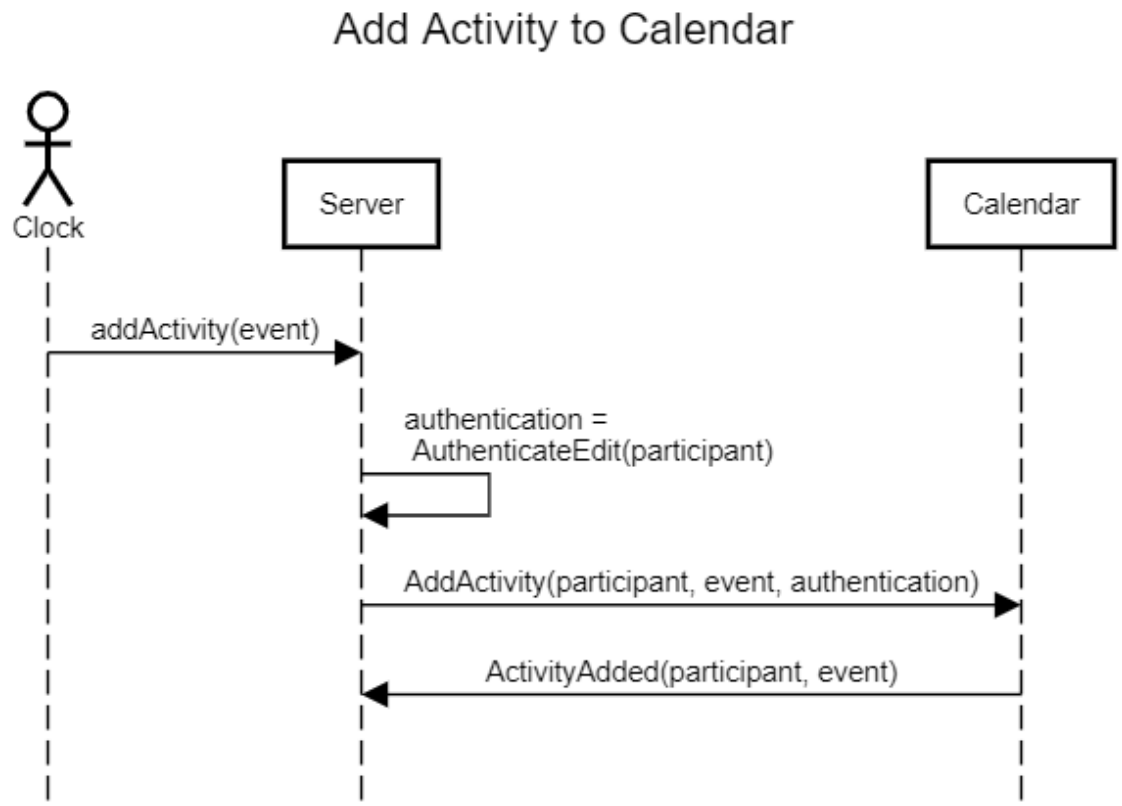


Figure 10: Add Activity to Calendar Sequence Diagram

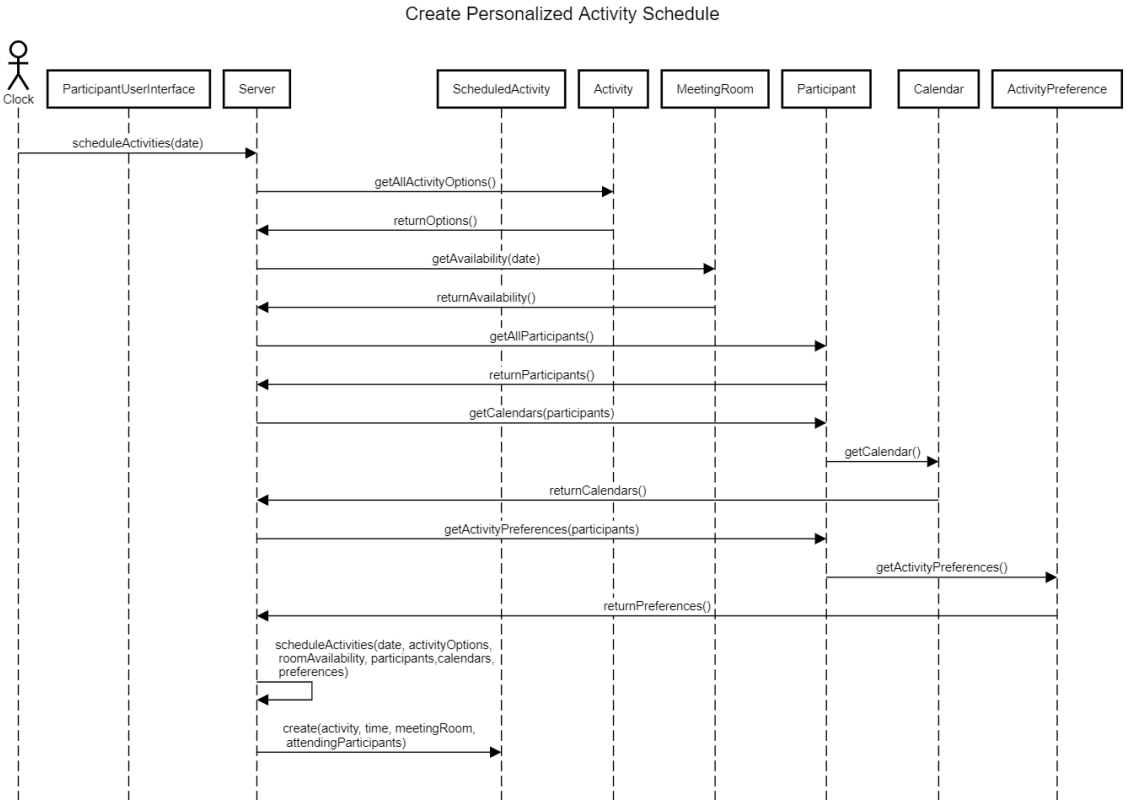


Figure 11: Create Personalize Activity Schedule Sequence Diagram

Class Diagram

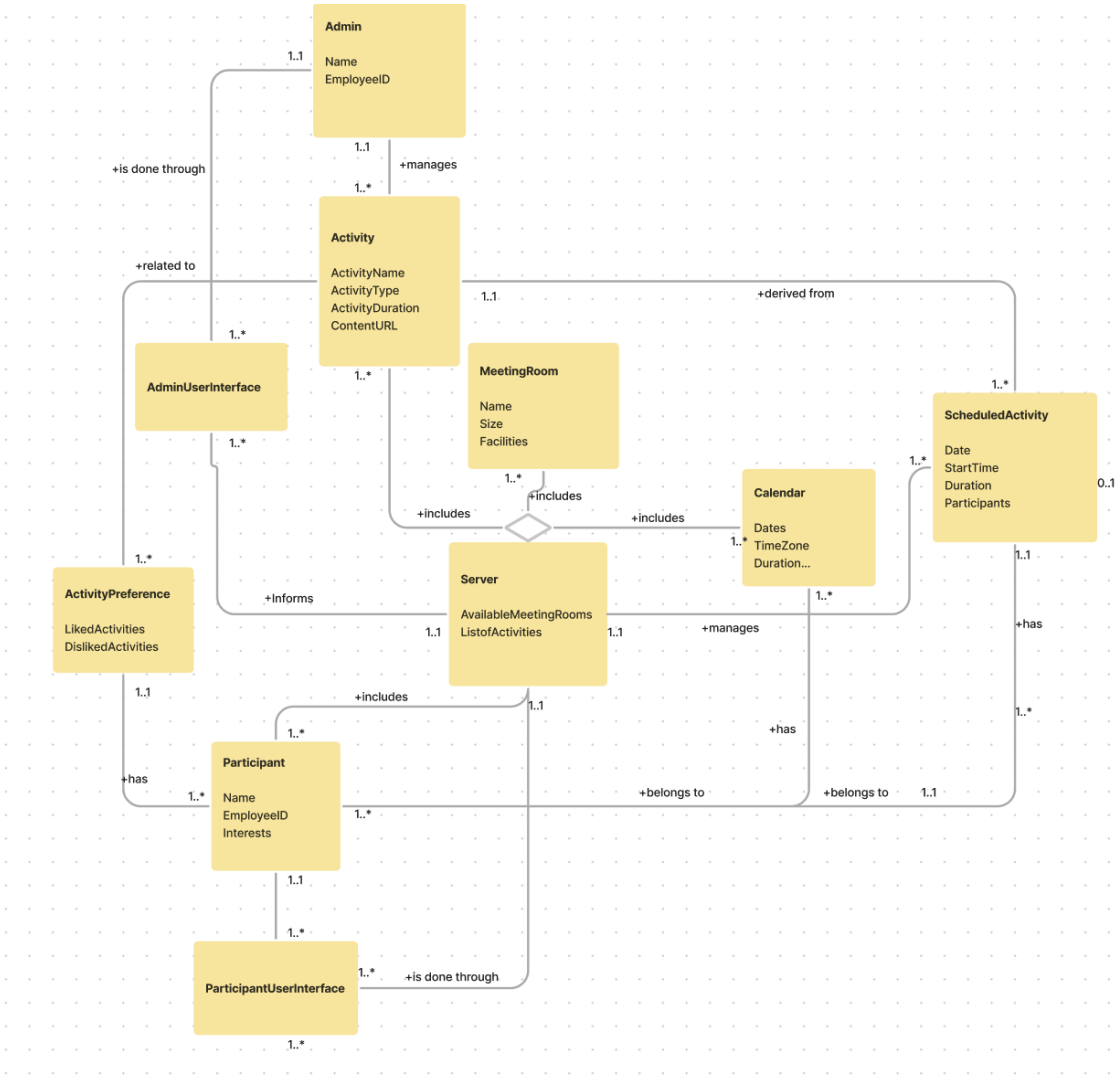
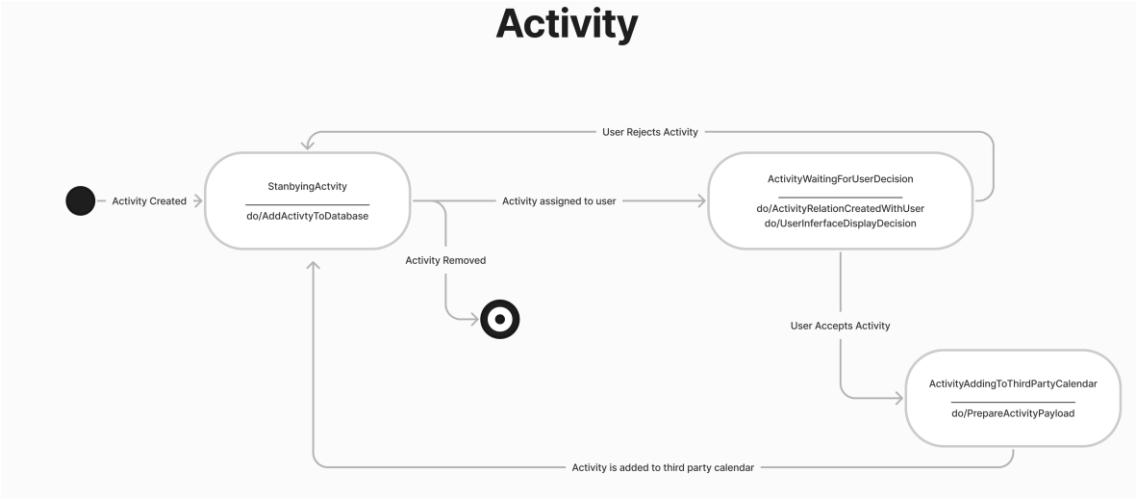


Figure 12: Phase 1 Class Diagram

Transition State Diagram



Fiigure 13: Activity Transition State Diagram

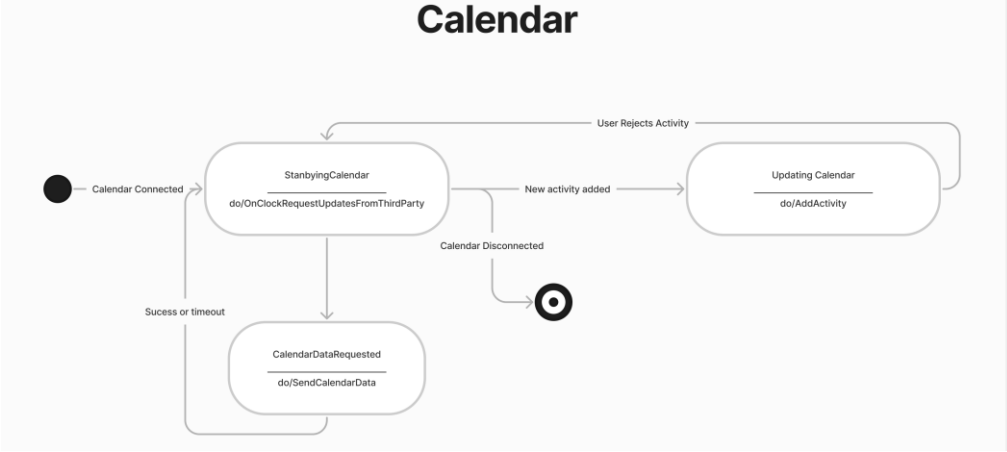


Figure 14: Calendar Transition State Diagram

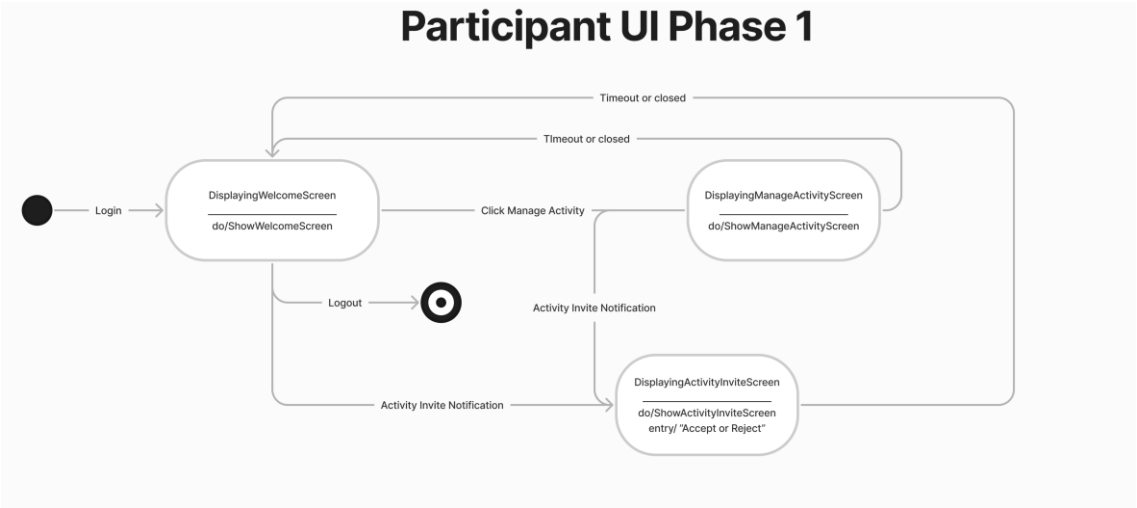


Figure 15: Participant UI Phase 1 Transition State Diagram

9.2Second Iteration

Second Cycle

9.2.1 Second iteration analysis

Expanded format use cases description

Use Case: Send User Activity Video
Actors: Clock
Purpose: To send the users the activity videos
Overview: When it is 30 minutes before the scheduled activity, the activity video gets sent to the Break Buddies website. Before the scheduled activity, the video will be hidden from users. When it is time for the activity to start, there will be a countdown and users will get access to the video. The video will play automatically. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.
Type: Primary
References:

Actor	System
1. The clock fetches activity content at scheduled time.	2. The server gets the content from the URL and sends to participants through user interface rendering.

Alternative Courses:
2. If the content is unavailable, an error is shown

Use Case: View Activity Invite
Actors: Employee
Purpose: To let employees views their invites to activities
Overview: An employee will visit the “Activity Invites” page through their account portal to view a list of new or unanswered invites to activities. New invites are indicated with a colored dot next to the tab name. For each activity in the list, they can view activity details by clicking on the activity. A modal will appear with a more detailed description, number of participants, and time/date. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.
Type: Primary
References:
Typical Course of Events:

Actor	System
1. The employee views the activity invite details by clicking on the activity.	2. The server renders the invites per participant and marks it as read.

Alternative Courses: N/A

Use Case: Respond to Activity Invite
Actors: Employee
Purpose: To let employees respond to their activity invites
Overview: On the page, "Activity Invites", employees can respond directly on the list or in the modal. They can respond by selecting 'Yes', 'No', or 'Maybe' for each activity. If 'Yes' or 'Maybe' is selected, the activity will be added to their calendar. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.
Type: Primary
References:

Actor	System
1. The employee responds to the activity invite by clicking 'Yes', 'Maybe', or 'No' in the modal.	2. The system will add the activity to the participant's calendar and their according response ('Yes' or 'Maybe'). The participant will also be added to the list of participants.

Alternative Courses:
2. An error is shown if the activity cannot be added to the calendar
2. The system will not add the activity to the calendar if the employee responds 'No'

Use Case: Watch Video for Activity
Actors: Employee
Purpose: For employees to collectively participate in a break activity
Overview: When it's time for the break, an embedded video will be shown on the home page and activities page. Employees can start playing the video and start following the guided activity. The employee can also pause and scrub the video. The interface is responsive, error-free, and consistent with TD branding. The user can perform the action with low latency on any operating system/browser. Communication happens through a private and encrypted web connection to the system server hosted on AWS.
Type: Primary
References:
Typical Course of Events:

Actor	System
1. Employee is on the home or activities page and starts the video at, or after, the scheduled time.	2. The web page will display and play the video.
3. Employee interacts with the video by pausing.	4. Video will stop playing.

Alternative Courses:
3. Employee can skip to an indicated timestamp

Conceptual Model

For this iteration, the conceptual model is a class diagram. It is shown in Figure 20.

Operation contracts

Name: showInvites()
Responsibilities: Show the scheduled activities the user is invited to in the user interface.
Cross-referencing:

- System functions: UF-001
- Use case: View Activity Invite

Notes: N/A
Exceptions: N/A
Output: None
Pre-conditions: The user has been invited to at least one scheduled activity.
Post-conditions: The activity invite is visible in the user interface.

Name: inviteResponse(answer: Boolean)
Responsibilities: Accept or reject an invite to an activity on the user interface. If accepted, add the activity to the user's calendar.
Cross-referencing:

- System functions: UF-009
- Use case: Respond to Activity Invite

Notes: N/A
Exceptions: An error should be thrown if the activity cannot be added to the calendar.
Output: An external request to the user's calendar app to add the activity
Pre-conditions: The user has received an activity invite on the user interface.
Post-conditions: If accepted, the scheduled activity is added to the user's calendar.

Name: fetchActivityContent(activity: ScheduledActivity)
Responsibilities: At the time of the activity, fetch the content for the activity and render it in the user interface.
Cross-referencing:

- System functions: UF-001
- Use case: Send User Activity Video

Notes: N/A
Exceptions: An error should be thrown if the content is unavailable.
Output: The URL for the content is sent directly to users via email.
Pre-conditions: The user has accepted a ScheduledActivity, and the activity is beginning.
Post-conditions: The user interface contains the activity content for viewing.

Name: startVideo()
Responsibilities: Begin a Scheduled Activity Video for a user once the activity start time is reached and the user prompts it.
Cross-referencing:

- System functions: UF-009
- Use case: Watch Video For Activity

Notes: N/A
Exceptions: N/A
Output: None
Pre-conditions: The user has accepted an activity invite, the activity is in the user's calendar app, and the activity's start time has commenced.

Post-conditions: The video begins playing in the user interface.

Name: interactWithVideo(stopped: Boolean, goToTime: minutes/seconds)
Responsibilities: Performs user interactions with a video, such as pausing and playing as well as skipping to an indicated timestamp.
Cross-referencing:

- System functions: UF-009
- Use case: Watch Video For Activity

Notes: N/A
Exceptions: N/A
Output: N/A
Pre-conditions: The activity has begun, and the user has started the activity.
Post-conditions: The video stops or resumes play, or skips to the indicated timestamp.

9.2.2 Second iteration Design

Sequence diagrams

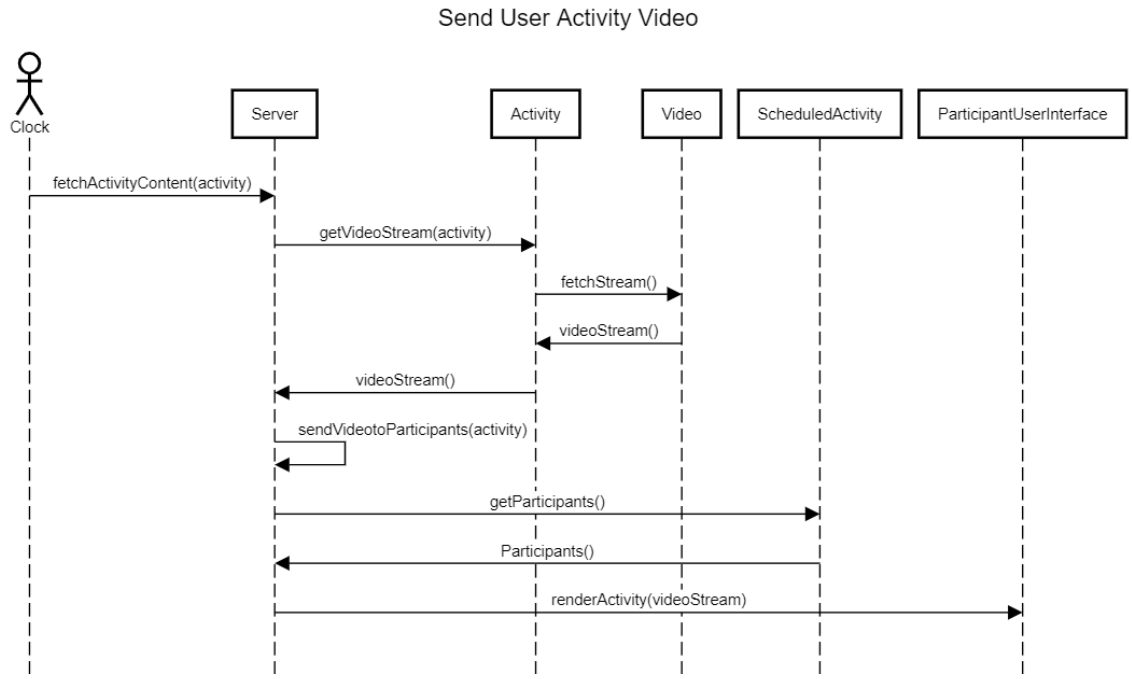


Figure 16: Send User Activity Video Sequence Diagram

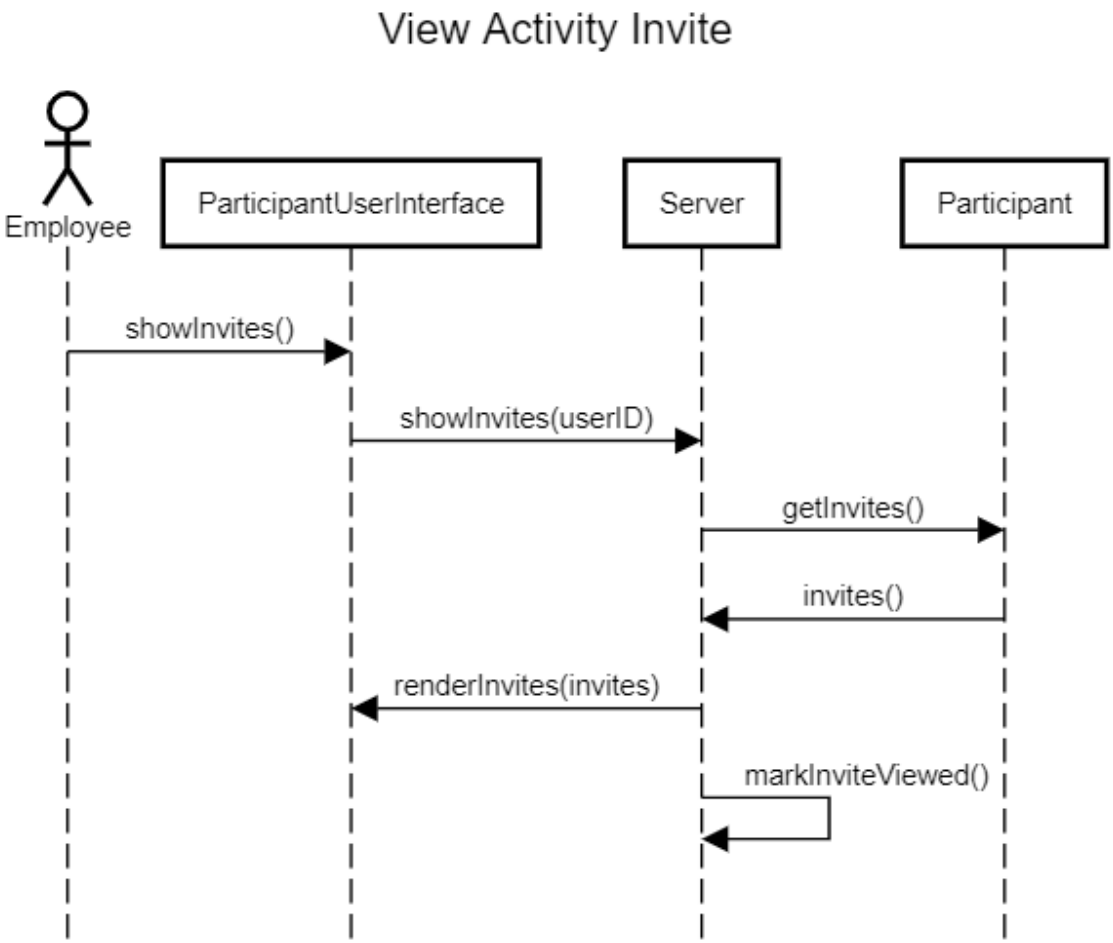


Figure 17: View Activity Invite Sequence Diagram

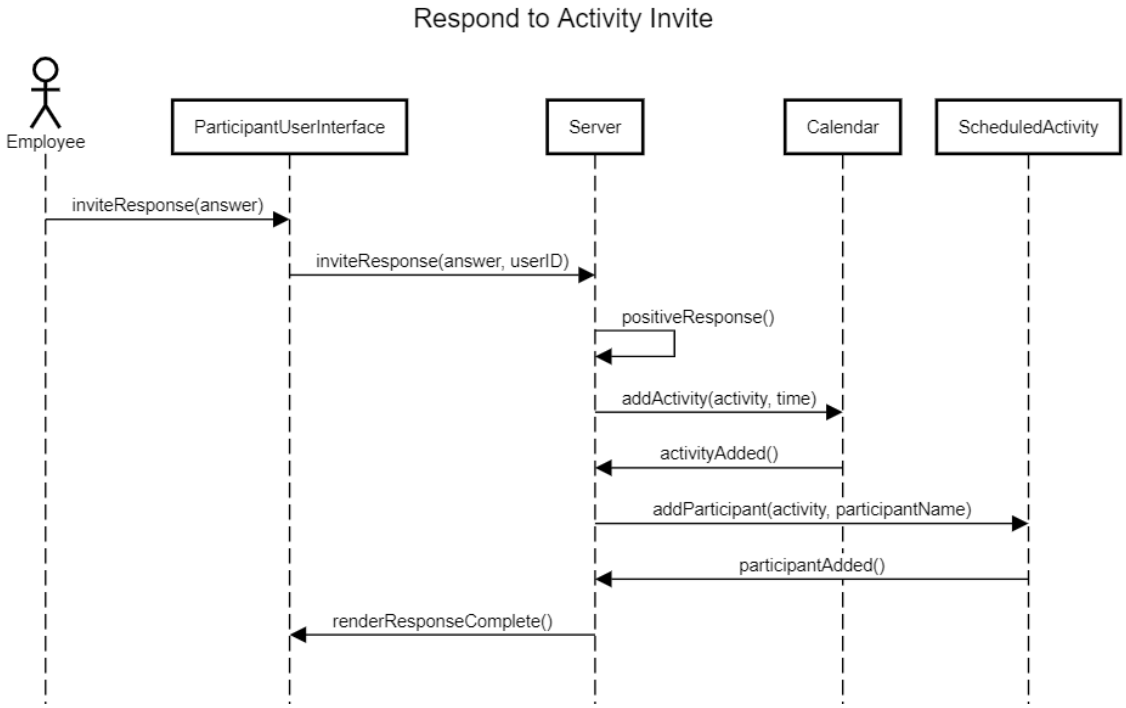


Figure 18: Respond to Activity Invite Sequence Diagram

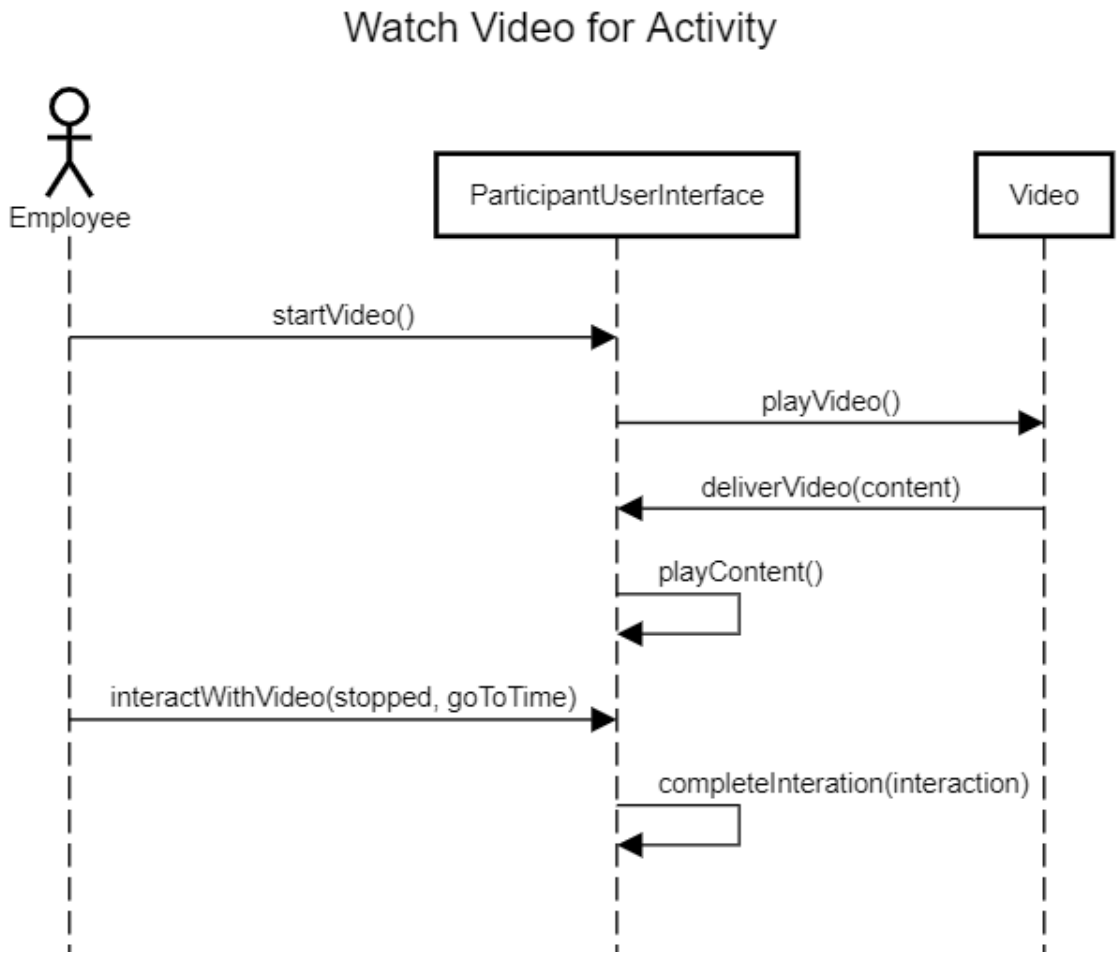


Figure 19: Watch Video for Activity Sequence Diagram

Class Diagram

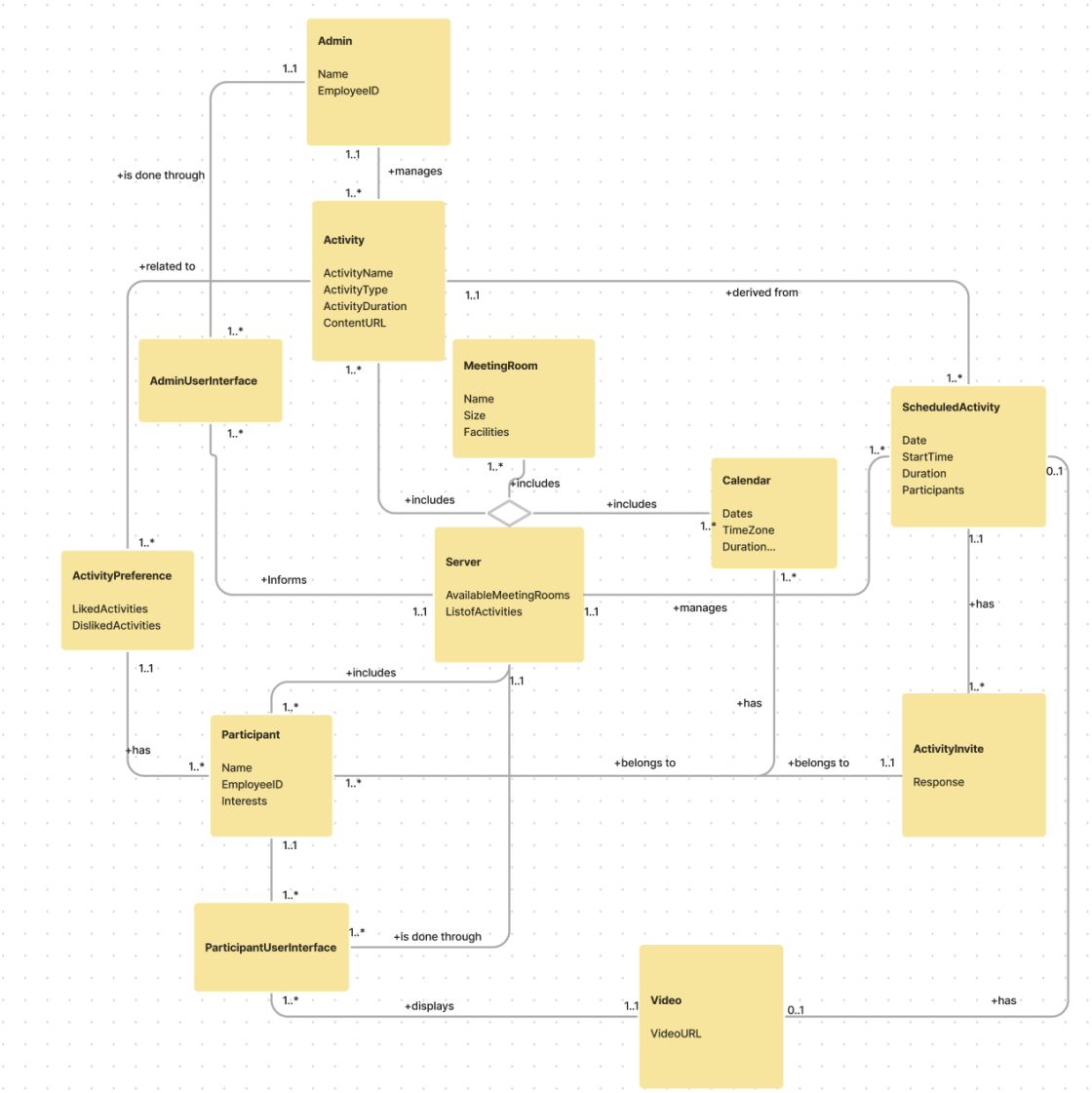


Figure 20: Phase 2 Class Diagram

Transition State Diagram

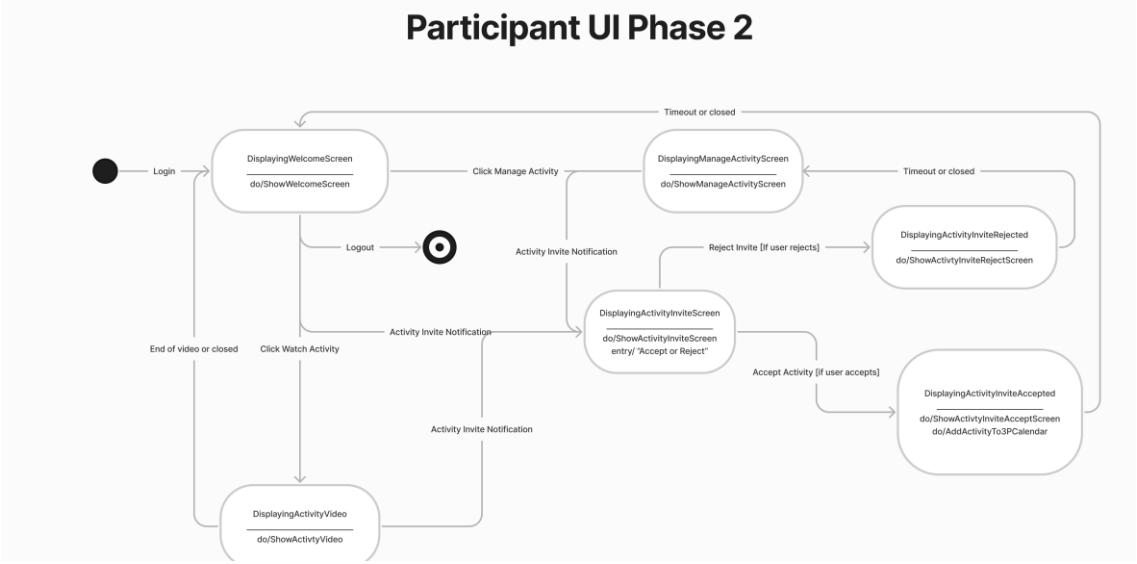


Figure 21: Participant UI Phase 2 Transition State Diagram

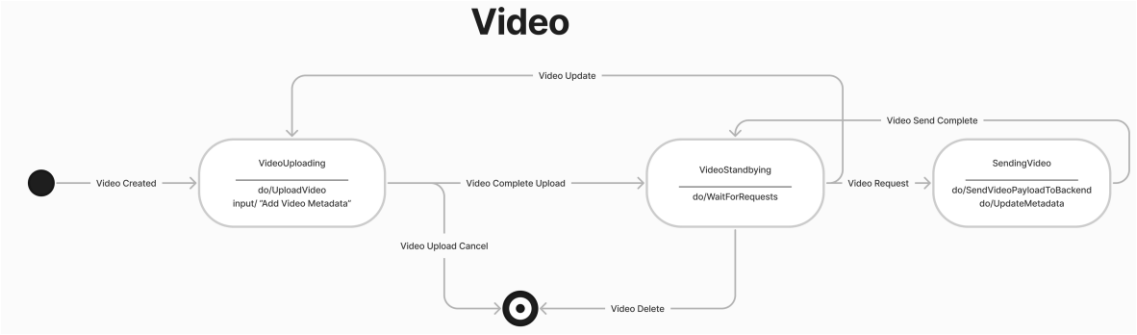


Figure 22: Video Transition State Diagram

10. Execution of the quality plan

See section 4.10

11. Execution of the configuration management plan

See section 4.10