2. Change Report

Group Number: **Team 20**

Group Name: **Gourdo Ramsay**

Group Members:

Lauren Waine
Megan Bishop
Tommy George
Bartek Grudzinski
Davron Imamov
Nathan Sweeney

2a)

We were tasked with modifying the Assessment 1 deliverables, documentation, and code that we inherited from the previous team. Our objective was to effectively plan, implement, monitor, and evaluate these changes to ensure a positive outcome. This process required crystal-clear communication, effective collaboration, and the efficient application of instruments to expedite our work.

To begin, we discussed our respective duties and responsibilities within the undertaking. This was essential for establishing precise expectations and ensuring that each team member understood his or her specific contribution to the project.

Then, we utilised Discord as our primary means of communication, which allowed us to schedule meetings and discuss the second portion of the assessment in depth. The voice and text messaging features of Discord made it simple for us to communicate in real time, pose questions, and exchange ideas. Regular meetings were scheduled to keep everyone informed and involved in the decision-making process.

We resorted to Google Drive for planning and documentation. This platform enabled us to store and manage documents in a central location that was accessible to the entire team. Google Docs was used to create and update documentation, while Google Sheets was used to monitor progress and manage deadlines. These tools provided a collaborative environment in which we could simultaneously modify documents and leave clarification or feedback comments.

When it came to collaborating on code, we selected GitHub as our principal platform. GitHub was a comprehensive platform for version management and code review. We used the pull request feature to coordinate code modifications, ensuring that all revisions were reviewed by at least one other team member before being merged into the main branch. In addition, we utilised GitHub Actions to automate testing, ensuring that the quality of the code was maintained throughout the duration of the project.

We utilised Trello, a web-based list-making application, to keep everyone informed of programming duties and progress. The intuitive interface of Trello allowed us to organise and prioritise tasks using boards, lists, and cards. This enabled us to maintain a comprehensive view of the project, monitor our progress, and ensure that all tasks were assigned and completed.

We used PlantUML to generate UML class diagrams for the purpose of visualising our ideas and designs, which helped us comprehend the structure and relationships of our code. In addition, we utilised Gantt charts to effectively plan our project's timeline and visualise task dependencies.

We maintained a healthy balance between flexibility and accountability throughout the process. We were aware that deadlines could shift due to unanticipated obstacles or alterations in requirements. However, we were accountable for meeting these deadlines as precisely as possible.

2b)i)

This report evaluates the new requirements added to the game developed by “Lucky” Team 13. These new requirements were added to improve the functionality and entertainment of the game. The requirements were implemented based on research from the book “Software Engineering” by Ian Sommerville [2].

A significant addition to the game was the incorporation of pizza and jacket potato. To enable these recipes, new ingredients were added such as, rolled dough, sauce, cheese, and pizza toppings were added. Additionally, two new stations were added [3], a baking station for cooking pizza and an unlockable station that can only be unlocked by accumulating points and purchased in the shop.

To add more challenge to the game, a time limit was introduced for completing orders. Players must complete each order within the allocated time or risk losing reputation points. A total of three such losses results in an overall loss of reputation.

In addition, an Endless Mode was added, allowing players to play the game without a specific objective [4]. To simulate actual restaurant scenarios and make the game more engaging, a menu system, customer counter, and customer groups were added. In addition, five power-ups were introduced to assist players advance and progress levels more quickly, enhancing the entertainment of the game.

To improve the overall game experience, the existing code was edited. Chefs were made to remain at their respective stations, and carrying stacks were included instead of one item. Various customer intervals were incorporated, and the choice of difficulty levels was provided [5].

Lastly, save and continue requirements were added to the game, allowing players to resume their progress at a later time. It is noteworthy that some old requirements were retained because they turned out to be useful and crucial for the system to run smoothly. To ensure that the new requirements align with the original requirements, the current requirements were linked to the previous report. The original report can be accessed at [1].

The team’s commitment to quality control and effective project management ensured that the new requirements were seamlessly integrated into the existing system and met the expectations of all stakeholders.

Assessment 2 requirements:

User Requirements:

ID	Description	Priority
UR_ENDLESS_MODE	A constant amount of customers come until you lose.	Shall
UR_INVEST_MONEY	Can spend earnt money on upgrades	Shall
UR_UNLOCK	Can unlock more stations, recipes or staff	Shall
UR_SAVE_GAME	Save game	Shall
UR_CONTINUE_GAME	Continue Game	Shall
UR_CARRYING_STACK	The player may be able to carry a stack of items at once, instead of just one item at a time, to serve customers more efficiently	May
UR_CHALLENGE	The game should provide an appropriate level of challenge to the player to keep them engaged and motivated	Should
UR_INSTRUCTIONS_PAGE	The game shall provide a instructions page for the player to follow	Shall
UR_REALISM	The game shall provide a realistic customer flow by varying the time intervals between customer arrivals	Shall
UR_EFFICIENCY	The player shall be able to organise tasks in a way that maximises their efficiency and reduces waiting times for customers	Shall
UR_CUSTOMISATION	The player shall be able to customise their task organisation to fit their personal playing style	Shall
UR_CUSTOMER_LEAVES	Customers may leave if they wait too long or if they receive poor service	Shall
UR_REPUATION	The game should track the player’s reputation, which can be improved or damaged	Shall

	based on customer satisfaction	
UR_SCENARIO_MODE	A mode in which the player must complete a series of levels or challenges with increasing difficulty	May
UR_LOSE_GAME	The game should have a way for the player to lose, such as by running out of money or failing to meet customer demands	Shall
UR_ENDLESS_SCORE	The game should keep track of the player's score as they serve customers, even in endless mode	May
UR_FOOD_PREP	The player must be able to prepare food items for customers, such as chopping vegetables or grilling meat	Shall
UR_RECIPE	The game should include multiple recipes for the player to learn and master	Shall
UR_INGREDIENT_DOUGH	The player must have access to ingredients to add cheese to pizzas	Shall
UR_INGREDIENT_BEANS	The player must have access to ingredients to make bean-based dishes, mainly the jacket potato	Shall
UR_INGREDIENT_SAUCE	The player must have access to ingredients to make pizza sauce	Shall
UR_INGREDIENT_CHEESE	The player must have access to ingredients to add cheese to pizzas	Shall
UR_PIZZA_TOPPINGS	The player must have access to pizza toppings to add to pizzas	Shall
UR_WIN	The game must be won when the user has finished serving all the customers	Shall
UR_CHEF	All chefs should be unique	Shall
UR_INGREDIENT_POTATO	The player must have access to ingredients to make potato-based dishes, mainly the jacket potato	Shall

UR_RECIPE_PIZZA	The game should include recipes specifically for making pizza, which may include different crust types, sauce options, and toppings	Shall
UR_STATION	The player must be able to manage different stations in the restaurant, such as the pizza oven, the prep area, and the fryer	Shall
UR_GAMEPLAY	The game should be fun and engaging to play, with a variety of tasks and challenges to keep the player interested	Should
UR_DIFFICULTY	The game should provide a range of difficulty levels to accommodate players of different skills levels	Should
UR_POWERUP	The player may be able to earn power ups or bonuses that can help them serve customers more efficiently or increase their score	May
UR_LEADERBOARD	The game should include a leaderboard where players can compare their scores and achievements with others	May
UR_FOOD_QUALITY	The quality of the food prepared by the player should affect customer satisfaction and reputation	Should
UR_MONEY	The game should allow the user to make money on the completion of an order based on the speed of completion	Shall
UR_TIME	The game should be be fast paced and the time to complete an order should not be too long	Shall
UR_CONVENTIONS	The game should follow standard conventions	May
UR_COLLISION	The game should not allow the chefs to phrase through each other	May
UR_INACTIVITY	The game should play a pre-recorded demo when the user is inactive for a given time	Shall

Functional Requirements:

ID	Description	User Requirements
FR_CUSTOMER_GROUPS	The system shall increase the odds of customers arriving in pairs or groups of 3 as the game progresses	UR_CUSTOMER_ARRIVAL
FR_CUSTOMER_TIMES_OUT	The customer shall leave after waiting too long and a reputation point shall be deducted	UR_CUSTOMER_LEAVES UR_REPUTATION
FR_CUSTOMER_INTERVALS	The system shall randomly generate a time interval between customer arrivals, with some intervals shorter or longer than others, to add minor challenge to the game	UR_CUSTOMER_ARRIVAL UR_CHALLENGE
FR_SCENARIO_LOSE	The system shall award the player with a loss if rep points reach 0	UR_SCENARIO_MODE UR_LOSE_GAME
FR_ENDLESS_MODE	The system shall infinitely send more customers to be served in increasing frequency	UR_ENDLESS_MODE
FR_INSTRUCTIONS_PAGE	The system shall display a instructions page in case the player is confused or needs new instructions	UR_INSTRUCTIONS_PAGE
FR_ENDLESS_SCORE	The system shall keep track of the amount of customers served and the highest number served in an endless run	UR_ENDLESS_MODE UR_ENDLESS_SCORE
FR_ENDLESS_END	The system shall end the endless mode when rep points reaches 0	UR_ENDLESS_MODE UR_LOSE_GAME
FR_PREP_FAIL	The system shall require a prep step is repeated should it be failed	UR_FOOD_PREP
FR_RECIPE_PIZZA	The system shall require that a pizza is made by through multiple steps	UR_RECIPE, UR_INGREDIENT_DOUGH, UR_INGREDIENT_SAUCE, UR_INGREDIENT_CHEESE, UR_PIZZA_TOPPINGS

FR_RECIPE_JACKET_POTATO	The system shall require that a jacket potato is made by through multiple steps	UR_RECIPE, UR_INGREDIENT_POTATO, UR_INGREDIENT_CHEESE, UR_INGREDIENT_BEANS
FR_INGREDIENT_DOUGH	The system shall provide a dough ingredient for the pizza recipe	UR_RECIPE_PIZZA
FR_INGREDIENT_SAUCE	The system shall provide a sauce ingredient for the pizza and jacket potato recipe	UR_RECIPE_PIZZA
FR_INGREDIENT_CHEESE	The system shall provide a cheese ingredient for the pizza and jacket potato	UR_RECIPE_PIZZA, UR_RECIPE_JACKET_POTATO
FR_PIZZA_TOPPINGS	The system shall allow for various pizza toppings to be added	UR_RECIPE_PIZZA
FR_INGREDIENT_POTATO	The system shall provide a potato ingredient for the jacket potato recipe	UR_RECIPE_JACKET_POTATO
FR_INGREDIENT_BEANS	The system shall provide a beans ingredient for the jacket potato recipe	UR_RECIPE_JACKET_POTATO
FR_STATIONS	The system shall provide cooking stations for the cooking stations	UR_STATION
FR_TIME_LIMIT	The system shall deduct reputation points if orders take too long to be served	UR_REPUTATION, UR_GAMEPLAY
FR_ENDLESS_DIFFCULTY	The system shall adjust the frequency of customer groups and order times based on the chosen difficulty	UR_ENDLESS_MODE, UR_DIFFICULTY
FR_MENU	The system shall provide a menu for customers to choose from	UR_GAMEPLAY
FR_BURGER_FLIPPING	The system shall allow the player to flip burgers	UR_GAMEPLAY
FR_CUSTOMER_COUNTER	The system shall display the number of customers served	UR_GAMEPLAY
FR_MONEY	The system shall provide in game money	UR_GAMEPLAY UR_INVEST_MONEY UR_MONEY
FR_SPEED_POWERUP	The system shall provide a	UR_POWERUP

	powerup that increases movement speed which can be bought with money	UR_INVEST_MONEY UR_MONEY
FR_COOKING_TIME_POWER UP	The system shall provide a powerup that increases movement speed which can be bought with money	UR_POWERUP UR_INVEST_MONEY UR_MONEY
FR_INSTACOOK_POWERUP	The system shall provide a powerup that allows instant cooking for 10 seconds which can be bought with money	UR_POWERUP UR_INVEST_MONEY UR_MONEY
FR_NO_BURN_POWERUP	The system shall provide a powerup that prevents burning during preparations which can be bought with money	UR_POWERUP UR_INVEST_MONEY UR_MONEY
FR_UNLOCK_ALL_POWERU P	The system shall provide a powerup that unlocks all stations or provides an extra chef	UR_POWERUP UR_INVEST_MONEY UR_MONEY
FR_REPUTATION	The player shall start with 3 reputation points	UR_REPUTATION
FR_LEADERBOARD	The system shall keep track of the highest scores and display a leaderboard table at the end of the game	UR_LEADERBOARD
FR_UNLOCK	The system shall provide the means to unlock more stations, recipes and staff	UR_UNLOCK UR_INVEST_MONEY UR_MONEY
FR_CARRYING_STACK	The system shall allow the player to carry a stack of items at once, instead of just one time at a time, to speed up serving customers	UR_CARRYING_STACK
FR_ORGANISE_BETTER	The system shall allow the player to organise the order in which tasks are completed to optimise their efficiency and serve customers more quickly	UR_EFFICIENCY UR_CUSTOMISATION
FR_CHEF_STATION	The system shall require the player to ensure the chef remains at the station until the food item has been fully prepared to prevent overcooking or burning.	UR_FOOD_QUALITY UR_REALISM
FR_FINISH	The system must recognise when all of the customers have been served	UR_WIN UR_LOSS

FR_MOVEMENT	The user must be able to move to the chef	UR_PLAYABLE
FR_CONTROLS	The controls decided should be accessible and easily understood even by people who aren't familiar with videogames	UR_PLAYABLE UR_CONVENTION
FR_SWITCH	The system must allow the use to switch chefs	UR_PLAYABLE
FR_INTERACT	The system must let the user interact with the kitchen	UR_STATION UR_PLAYABLE
FR_CONTINUE_ACTION	The system will allow the non-controlled chef to continue their previous action. Eg, continue chopping	UR_PLAYABLE UR_CHEF
FR_DEMO	The system will play a previously recorded gameplay that serves are the tutorial if left inactive for a given amount of time	UR_INACTIVITY
FR_COLLISION	The system will not allow the chefs to phrase through each other	UR_COLLISION UR_PLAYABLE

Non-Functional Requirements:

ID	Description	User Requirement	Fit Criteria
NFR_UNLOCK	The system shall be secure	UR_UNLOCK	The system shall provide a secure authentication mechanism for buying more staff based off the in-game currency
NFR_TIME_LIMIT	The system shall be time-sensitive	UR_REPUTATION UR_GAMEPLAY	The system shall have a time limit for completing orders
NFR_SCENARIO_ENDLESS	The system shall be scalable	UR_ENDLESS_MODE	The system will support an appropriate number of customers
NFR_MENU	The system shall be user-friendly	UR_GAMEPLAY	The system shall have a user-friendly menu that lists available items and options

NFR_STATIONS	The system will be modular	UR_STATIONS	The system shall have separate stations for baking and unlocking the shop, which must be modular and secure
NFR_POWERUPS	The system will be efficient	UR_POWERUP UR_INVEST_MONEY UR_MONEY	The system shall have five power-ups that increase speed, reduce cooking time, instacook for 10 seconds, no burn, and unlock all stations or extra chefs.
NFR_DIFFICULTY	The system will be adjustable	UR_DIFFICULTY	The system shall have different difficulty levels that change the odds of groups and order times
NFR_SAVE_GAME	The system will be persistent	UR_SAVE_GAME	The system shall allow users to save their progress
NFR_REPUTATION	The system will be accountable	UR_REPUTATION	The system shall deduct three reputation points for taking too long to complete an order, and all reputation points for losing overall.
NFR_CONTINUE_GAME	The system will be resumable	UR_CONTINUE_GAME	The system shall allow users to continue their progress
NFR_RESPONSIVE	The system will be responsive	UR_PLAYABLE	The response time should be less than one second after input
NFR_AVAILABLE	The system will be able run and not crash	UR_PLAYABLE	Uptime: 99% during game time

Constraint Requirements:

ID	Description
CON_PROGRAMMING_LANGUAGE	The system must be written in Java (version 11).
CON_OS	The system must be able to run on all desktop operating systems: Linux, Windows and MacOS

CON_SYS_REQUIREMENTS	The game must be able to run on any modern desktop
CON_RESOLUTION	The game must adapt to any screen size

References:

[1] “Requirements v2”, Team13ENG1, 2022-2023. [Online].Available: https://team13eng1.github.io/files/assessment/Requirements%20v2.pdf. [Accessed: May 1,2023].

[2] Somerville, I.(2016). Requirements Engineering. In Software Engineering (10th ed., pp. 81-118). Pearson.

[3] Somerville, I.(2016). Requirements Elicitation. In Software Engineering (10th ed., pp. 119-150). Pearson.

[4] Somerville, I.(2016). Requirements Specification. In Software Engineering (10th ed., pp. 151-188). Pearson.

[5] Somerville, I.(2016). Requirements Validation. In Software Engineering (10th ed., pp. 189-226). Pearson.

2b)ii)

The state diagram for the game logic has been updated to accommodate the endless gamemode. Previously the acceptance state would have been reached when the last customer leaves. In the updated version of the game, if the player is not in the endless gamemode the final customer leaving still reaches the acceptance state. However if the endless gamemode is being played, the game will reach an acceptance state if the reputation variable reaches 0: otherwise the game will continue.

Team 13 State Diagram V2
https://team13eng1.github.io/files/assessment/State%20Diagram%20v2.pdf

Team 20 State Diagram
https://github.com/GourdoRamsay/GourdoRamsay.github.io/blob/main/files/team20/stateDiagram.jpg

The sequence diagram for the game logic has also been updated to accommodate the increased number of chiefs as well as the customers no longer waiting indefinitely. There are now three chiefs in the beginning, followed by support for an unlimited number of more chiefs: which the player can gain through the in-game store. Each customer now waits a given time limit before they step away from the restaurant and reputation is lost.

Team 13 Sequence Diagram V3
https://team13eng1.github.io/files/assessment/Sequence%20Diagram%20v3.pdf

Team 20 Sequence Diagram
https://github.com/GourdoRamsay/GourdoRamsay.github.io/blob/main/files/team20/sequenceDiagram.jpg

The chef attributes have been greatly reduced through recompiling the code and rewriting specific features. For example, the chef skin is no longer hardcoded through attributes for each ingredient, but instead just using skinNeeded. This has been the primary factor behind the decreased complexity of the chef class. This was done to declutter the code which makes the game more modular and prevents spaghetti code - code in which actions have undesired consequences because of poor design.

Team 13 Chef Attributes
https://github.com/GourdoRamsay/GourdoRamsay.github.io/blob/main/files/team13/documents/All_attributes_in_chef.pdf

Team 20 Chef Attributes
https://github.com/GourdoRamsay/GourdoRamsay.github.io/blob/main/files/team20/chefAttributes.png

Inside of the play station class due to the new requirements, many new attributes have been added. Specifically, the number of chefs is now unlimited which has introduced new attributes such as chefList and checCount. While removing attributes chef1 and chef2 in

order to make the game more modular. The changing requirements also led to the creation of the stations such as choppingBoards, pans and plateStations. This was done to meet the requirements of the client.

Team 13 Play Screen Attributes
https://github.com/GourdoRamsay/GourdoRamsay.github.io/blob/main/files/team13/documents/All_attributes_in_playScreen.pdf

Team 20 Play Screen Attributes
https://github.com/GourdoRamsay/GourdoRamsay.github.io/blob/main/files/team20/playScreenAttributes.png

The use case diagram has been updated to accommodate the new game logic. Specifically the endless mode in which there is no last customer. The customer can now leave if their timer runs out.

Team 13 Use Case Diagram V2
https://github.com/GourdoRamsay/GourdoRamsay.github.io/blob/main/files/team13/documents/Use-Case_Diagram_v2.pdf

Team 20 Use Case Diagram V2
https://github.com/GourdoRamsay/GourdoRamsay.github.io/blob/main/files/team20/useCaseDiagramV2.png

2b)iii)

Team 13 used Scrum which is a type of Agile development, our team decided to continue with our original development style of Agile, Extreme Programming (XP). XP was a better fit for our team's working style as it fit our hierarchy mentality of having no hierarchy. Since Scrum would require a Scrum master we decided to reduce the overhead by simply only having the programmers. Due to the scope of the project and the little number of programmers, this best suits our team. The team has a collective code ownership with the code being guided by a test-driven development. Although we did not use pair programming exactly, we did implement the programmers working together with one writing code and another helping them.

Team 13 used SmartDraw which is a web-based diagramming tool, we decided not to use this because our team members had no experience with the tool. They could achieve the same results using different tools, which therefore overshadowed the need for SmartDraw.

Team 13 used Trello to manage their tasks, we decided to use Trello too because it is an excellent tool for task management. It helps keep track of tasks and whom to assign them to, all while keeping track of the progression of the project.

Team 13 decided to complete the majority of the documentation before implementing had started, we decided to not follow this route. First of all, our programmers were used to digging into code before finalising the documentation. Furthermore, since the product is already built and needs expansion, we decided to give our programmers the freedom to explore the software in order to make better informed future decisions.

Team 13 had a Project Owner and Product Owner, likewise with the lack of a hierarchy in our team structure, we decided not to follow suit. The stakeholders acted as our Product Owner and we individually came together to act as the Project Owner. The freedom and sense of ownership over one's work encouraged each member of the team, individually we felt our collaboration had impact by having reliance on each other: having to report not to one individual but to one entity, the team.

Team 13's Method Selection and Planning
https://team13eng1.github.io/files/assessment/Method%20Selection%20and%20Planning.pdf
Team 20's Method Selection and Planning (Excluding GANNT charts)
https://github.com/GourdoRamsay/GourdoRamsay.github.io/blob/main/files/team20/msapTeam20.pdf

2b)iv)

Upon assuming control of the project that we inherited from “Lucky” Team 13, we acknowledged the significance of identifying potential risks and executing mitigation strategies to guarantee the achievement of project objectives. It is acknowledged that unanticipated risks and challenges may emerge throughout the duration of the project, and if not effectively managed, may result in project setbacks, heightened intricacy, or potentially project termination. Our risk identification and mitigation strategy was based on the “Software Engineering” book by Ian Sommerville [1].

Various identification and analysis techniques were employed to obtain an in-depth knowledge of potential risks [3]. The initial step involved a comprehensive examination of the project documentation, including the project plan, requirements, design documents, and test cases [2]. This aided us in gaining an understanding of the extent of the project and identifying potential areas of risk. Furthermore, an assessment was conducted to determine the potential consequences of the risks mentioned above on the project, considering variables such as probability and potential impacts.

Subsequently, we engaged in consultations with the appropriate parties to gather additional perspectives and detect any potential risks that might have been disregarded during the primary evaluation [4]. It was deemed imperative to engage with both internal and external stakeholders to obtain a holistic comprehension of their requisites, apprehensions, and anticipations. Feedback was also obtained from the preceding team, who offered valuable insights on the potential risks they had identified.

Upon conducting a comprehensive analysis of the risk report submitted by the previous team, it was determined that certain risks identified therein should be maintained [2]. By maintaining their risks, we guaranteed that our risk management methods considered all conceivable risks that may arise throughout the project’s lifespan. The mitigation strategies implemented by the preceding team were assessed to ascertain their continued relevance and efficacy.

Ultimately, a thorough risk identification procedure was employed to assess potential risks that may emerge throughout the duration of the project, and suitable methods for mitigation were identified [3]. The previously described process entailed the identification of plausible risks, a thorough analysis of their potential ramifications on the project, and the development of appropriate measures to alleviate their effects. The efficacy of said strategies was also evaluated to ascertain their suitability and productivity [5]. The original Risks and Mitigation report for “Lucky” Team 13 can be found at [6].

ID	Type	Description	Likelihood	Severity	Mitigation	Owner
Technology						
R1	Project	Poor documentation or incomplete understanding of the project's architecture	L	H	Perform a comprehensive review of the available documentation, and reverse-engineer the project to gain a deeper understanding of its architecture	Meghan
R2	Product	Unforeseen issues with existing code or dependencies	H	H	Conduct a comprehensive code audit and dependency review, and make necessary updates and upgrades.	Tommy
R3	Business	The game is incomplete or low quality	L	H	Have good plans and learn from mistakes from previous sprints	Bartek
R4	Project	Insufficient testing of the previous team's work	M	H	Conduct comprehensive testing of the previous team's work, and perform regression testing after making any updates	Lauren

R5	Product	Inconsistent or poor quality code	M	H	Establish and enforce coding standards and best practices, and perform code reviews on all code changes	Nathan
R6	Project	Resistance to change or lack of buy-in from team members	L	M	Communicate the benefits of the changes to the team, and involve them in decision-making processes throughout the project	Davron
			People			
R7	Project	Poor communication or collaboration between team members, leading to misunderstanding or conflicts	M	M	Maintain open communication with stakeholders and management, and be prepared to adapt to changes as necessary	Lauren
R8	Product	Poor user experience or design due to inadequate user research or testing	M	H	Conduct thorough research and testing throughout the development process, and incorporate feedback into the design	Nathan

R9	Project	A team member becomes unavailable temporarily or permanently	M	M	Have tasks assigned to at least two groups members and have code and documentation easy to understand so another team member could continue the work	Meghan
R10	Project	Team members cannot attend all meetings due to personal activities	H	M	Make sure we make use of the timetabled practicals and make notes of what was discussed and what needs to be done	Meghan
R11	Business	Lack of clear project goals or objectives, leading to confusion or delays	M	M	Clarify project goals and objectives with stakeholders and team members, track progress regularly, and encourage open communication and transparency	Davron

R12	Business	Inadequate risk management or contingency planning, leading to unexpected issues or delays	M	H	Develop a comprehensive risk management plan, identify potential risks and their impacts, establish contingency plans, and communicate any changes to stakeholders and team members.	Tommy
R13	Business	Inadequate risk management or contingency planning, leading to unexpected issues or delays	M	H	Develop a comprehensive risk management plan, identify potential risks and their impacts, establish contingency plans, and communicate any changes to stakeholders and team members	Tommy
Requirements						
R14	Business	Conflicting requirements from stakeholders	M	H	Establish a requirements prioritisation process that incorporates stakeholder input and decision-making to resolve conflicts	Davron

R15	Project	Inaccurate documentation of requirements	L	H	Implement a rigorous requirements documentation process that includes regular reviews to ensure completeness, accuracy, and consistency with stakeholder expectations	Lauren
R16	Product	Inconsistent or incomplete traceability of requirements	H	M	Develop and maintain a comprehensive traceability matrix that tracks requirements from inception through delivery, ensuring all requirements are linked to corresponding design and test artefacts.	Meghan
R17	Business	Poorly defined acceptance criteria	M	H	Collaborate with stakeholders to establish clear and measurable acceptance criteria that are achievable within project constraints	Bartek

R18	Project	Vague or ambiguous requirements	H	M	Utilise structured requirements elicitation techniques, such as use cases or user stories, to establish clear and unambiguous requirements, and involve stakeholders throughout the process to ensure clarity and completeness	Tommy and Nathan
Estimation						
R19	Business	Lack of stakeholder involvement in the estimation process	M	H	Involve stakeholders, such as project sponsors or business owners, in the estimation process, providing transparency and involving them in decision-making to ensure alignment with project goals and expectations	Lauren

R20	Product	Ambiguity or volatility in project requirements and scope	H	H	Break down work into smaller tasks and involve stakeholders in estimation to manage ambiguity and volatility in project requirements.	Meghan
R21	Project	Underestimation of the complexity of the project, including dependencies, risks, or technical challenges	H	H	Conduct thorough project analysis to document requirements and dependencies . Use expert judgement or historical data analysis to ensure estimates accurately reflect project complexity.	Bartek
R22	Business	Inadequate consideration of external factors, such as market conditions or regulatory requirements	M	M	Conduct a thorough analysis of external factors and incorporate them into the estimation process to ensure that the estimates accurately reflect the project environment	Tommy and Nathan

R23	Project	Inaccurate or incomplete data	M	H	Conduct a thorough data gathering process, involving stakeholders and experts as necessary, and implement appropriate statistical methods to analyse the data and identify trends and patterns	Lauren

References:

[1] Somerville, I. (2016). Risk Management. In Software Engineering (10th ed. Boston, MA). Pearson.

[2] Somerville, I. (2016). Risk Management. In Software Engineering (10th ed., pp. 399-428). Pearson.

[3] Somerville, I. (2016). Risk Management. In Software Engineering (10th ed., pp. 81-118). Pearson.

[4] Somerville, I. (2016). Risk Management. In Software Engineering (10th ed., pp. 323-358). Pearson.

[5] Somerville, I. (2016). Risk Management. In Software Engineering (10th ed., pp. 515-536). Pearson.

[6] “Risk Assessment and Mitigation v2”, Team13ENG1, 2022-2023. [Online].Available: https://team13eng1.github.io/files/assessment/Risk%20Assessment%20and%20Mitigation%20v2.pdf. [Accessed: May 1,2023].