Func Requirement: What does the system do?
Non-Func Requirement: How does the system do it?

Attribute	Attribute
Correct	Traceable
Justified	Delimited
Complete	Readable
Consistent	Modifiable
Unambiguous	Verifiable
Feasible	Prioritized
Abstract	Endorsed
Necessary, Attainable, Verifiable	

The 7 sins of the specifier: noise, silence, contradiction, ambiguity, wishful thinking, overspecification, and dangling reference.
Single Statement of Need: clear, concise statement about the system's overall goal and how it will accomplish those goals.
Elicitation Techniques: Background reading, interviews, questionnaires and surveys, and observing behaviour.
Personas are VARIED: vivid, actionable (does it help the team sell/build stuff?, real (go where the users are, observe, interact), identifiable (developers must wear the shoes of personas), exact, detailed

Inputs/outputs of an RE Process:
Inputs: Stakeholders (needs, goals), Information (existing system, domain), Regulations (internal, external). Outputs: Agreement (agreed requirements), Specification (user req, system req), Models (user/system).

Elicitation: Collecting info to identify problems and opportunities. Techniques: interviews, shadowing, etc.
Analysis: Building models of req. That can be used for validation. Identifying conflicts between req. Techniques: UML, SysML, tables, etc.
Specification: what users need to be able to do with the system, what the system must do for stakeholders, properties/qualities expected from system. Techniques: req. Matrices, template- guided approaches
Negotiation: dealing w conflicting requirements. Techniques: consensus building, majority rule, appeal to authority.
Validation: checking req. Doc to ensure correct system is built. Techniques: informal (reviews, demos, walkthroughs), semi-formal (prototyping, simulation), formal (formal proofs, model checking)
Documentation: recording valuable info, helps with long term.Techniques: controlled natural language, generation from models.
Types of Reqs: 1. Vision and scope reqs **2.** User reqs (for non- tech ppl) **3.** System reqs (technical).
Types of functional reqs: Transformation (required response to condition/event), invariant, and failure description.
Types of non-functional reqs: optimization, reliability/availability, timing, precision.
Safety Reqs: separately identified and managed, signed off by a regulatory body, focusing on identifying hazards.
Security Reqs: identified, managed, and signed off. Expressed by a security policy.
Good NF req has 2 parts: 1. description (capturing stakeholders' intentions) **2.** Fit Criterion (quantifying this intention).
Actor: ppl, sub-systems, or indirect stakeholders who act in a system to achieve a goal.
Types of Risks: time risk, budget risk, scope risk, external risk, and single point of failure (risk that has potential to be catastrophic).
Ways to mitigate risk: avoid it, accept it, reduce/control it, or transfer it.

Model: a simplified or idealized description or conception of a particular system, situation, or process, often in mathematical terms, that is put forward as a basis for theoretical or empirical understanding, or for calculations, predictions, etc.; a conceptual or mental representation of something.
A good model: reduces the amount of complexity, is inexpensive, facilitate description, facilitate the analysis (will that work?).
Example of analysis: informal model (templated text in req doc), semi-formal model (goal model), formal model (alloy model).
Enterprise Modelling: describing the behaviour of the norganization in which the system will operate. It covers **1.** Understanding the org's structure **2.** the goals of the org **3.** The business rules that affect its operation **4.** Tasks/responsibilities of members.
Domain Modelling: abstract description of the world in which the system will operate
Use case diagram: used to capture the system's functional (behavioural) reqs. **Describes** how an actor achieves a goal using a system. **Describes** how the system protects the interests of the stakeholders.
Primary actor: a stakeholder whos goal is met by the use case. **Supporting actors:** other actors who's participation is necessary to achieve the primary's goal.

Scenario: description of a particular sequence of actions or interactions. Captures a particular way of achieving a stakeholder goal. One particular path through a use case.
Main success scenario: represents the usual behaviour.
Secondary scenario: we can augment steps with alternative scenario (other action that could be done?), or with exception scenario (something that can go wrong? aka ways to fail).
Exceptional scenario:
- 3.1-8.1: customer to young; Sales Clerk refuses to sell tickets

Replaced Steps — Condition — step(s)

Use cases are good for: 1. modelling current work practices **2**. Modelling how a system ought to operate **3**. Uncovering error cases, pre/postconditions, scenario specific NF reqs, and reqs on supporting actors.
Use cases and scenarios are suitable to identify: cross-cutting concern, and silence.

Domain modelling:
Analysis: deriving acceptance tests from use cases, constraints from goals…
UML for reqs: use case diagrams, static diags, dynamic behaviour diags.
Class diagrams limitations: they don't help us organize requirements (b/c we are working on "business" concepts), working on smaller elements reinforces a sense of ownership.

UML: Unified **M**odelling **L**anguage
-described as an open standard by the Object management group.
-tooling is available independently from the standardization process.

WE ARE NOT MAKING DESIGN DECISIONS!!

Behaviour modelling:
Activity Diagram: describe a control flow between activities, useful when wanting to express concurrency triggered by different events. They combine activities from flowcharts, Petri nets, Workflow, and SDL state modelling. Share elements with state diags.
State Diagrams: model the lifecycle of an object, in terms of states and state transitions, based on Harel's state charts.

For endorsement: focus on the product owner for the business side and the tech lead on the development side.
NFRs: cannot be expressed as a function of the system, they can be regarded as a constraint on the system.
Taxonomies are exclusive and exhaustive. **Aspects** are not exclusive and might overlap, they can also conflict with each other.
Examples of aspects: functional, usability, international, performance/reliability, environmental, legal, safety and security, development aspect, deployment aspect, and maintenance aspect.
Usability: effectiveness, efficiency, learnability, memorability, satisfiability. **International aspect:** language, date format, and laws. **Environmental aspect:** location and physical envs, interoperability.
Performance: speed, capacity, accuracy, reliability, availability. **Legal aspect:** safety and security, data protection act, anti-discrimination act/equality/equity laws, and accounting laws.
Development aspect: process and tools to be used (eps/analysis process, design process, implementation, version control system), project management. **Deployment aspect:** shrink wrapped product, and bespoke installation (documentation, and support!!!). **Maintainability:** maintenance and portability.
Scope of NFRs: system wide (product req, process req, external reqs), single function NFR (associated with one given case).
Use cases NFRs: can be associated at multiple levels: - all use cases (factorized approach), individual use case, individual step of a given UC (When submitting card details they must be secure)
Approaches supporting validation: soft (reviews and inspections, prototyping and testing), hard (formal methods, model simulations). IT —> soft approach, safety-critical systems —> hard
Difficulty with inspections: getting management commitment, protecting resources (should we rush this feature or skip the inspection?), and getting engineer acceptance.
Methods of settling conflicts: cooperative methods (negotiation and education), competitive methods (combat coercion and competition), 3rd-party methods (arbitration, appeal to authority).
Prototyping: the process of building a working model of the system. **Types: 1. Throwaway:** discard after knowledge is gained **2. Evolutionary:** incrementally developed into real system.
3. Low fidelity: most often on paper using the "wizard of oz" approach. **4. High fidelity:** automated, eg. inVision, Figma…

SCALAR VALUES ARE NOT ACCEPTABLE FOR NFRS

Agile Manifesto: individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, responding to change over following a plan. *That is while is there is value in the items on the right side of "over", the items in the left are more important.*
1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable sobehavior2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
Scrum method: planning, implementation, review, and retrospective
Epic: a large user story that define a coarse-grained feature, does not come with an acceptance scenario, will be refined into multiple (INVEST) stories.

User story: is the description of value provided to a persona through an action. A triple (persona, value action).
Stories are INVESTment: Independent, negotiable, valuable, estimate, small, testable

Agile: MAXIMIZE VALUE, NOT OUTPUT

CRF: conditional random fields.

Security: YOU KNOW NOTHING (buy or reuse existing solutions) (Humans a the primary source of vulnerability)
Computer security: contextual and relative. Often comes as policies (considered a constraint or assumption). Often defined by SMEs.
Hinges on 3 things (**CIA**): **1. Confidentiality:** concealing information and resources **2. Integrity:** preventing improper change so that you trust the system. **3. Availability:** are the resources and data usable?
Security Policy: is a statement partitioning the states of a system into authorized and unauthorized states.
Confidentiality Policy: identifies states in which information can leak to individuals not authorized to recieve it.
Integrity Policy: specifies authorized ways in which information may be altered and which entities can be authorized to alter information. Relies on the *"Separation of Duties"* principle. If all entities in S trust R: $\forall s \in S$, trusts(s, R)
Availability Policy: describes what service must be provided (often comes with expected level of service). It also specifies high-level business rules (abstract).
Threat: a potential violation of security. Based on disclosure, deception, disruption, and usurpation.
Vulnerability: weakness in the system that makes it possible. Related to system and people that use the system.
Non-repudiation: you repudiate something if you deny ownership.
Security mechanism: an entity or procedure that enforces some part of a security policy.
Bell-LaPadula: relies on

RELAI CASE STUDY (Respectful & ExpLainable AI to support struggling people and mental health practitioners)

Mental Health care in Canada (pre-covid)

• 4.9M 15yo+ Canadiens needed mental health support (in 2015...)
- 1.6M felt that their needs were not met, or only partially
- 33% of Ontario's students (aged 12-17) reported the need to talk to someone
about their mental health (in 2017)
• Mental illness is a leading cause of disabilities in Canada
• Canada's specificities
- Not really a big country in terms of population: 38M, half of Italy, ~California
- A humongous territory (2nd largest inhabited territory in the World).

Stakeholders

• Principal Investigator: Dr. Marie-Jean Meurs
- Natural Language Processing, Artificial Intelligence
• Collaborators' expertise:
- Health: Psychiatrists (3), Digital health
- Social sciences: Online user be haviour, Philosophy, Ethics, Technology appropriation (2), Lawyer.
- Engineering: Privacy, Modelling

SSON: Helping practicians assess patients' status at the Psychiatric ER by using Natural Language Processing to evaluate patients' suicidal risk using their text messages from the last eight weeks.

Second SSON: Provide a frontline mental health support to Canadians as a mobile app that can detect suicidal risk and encourage users to consult a practician.

Confidentiality
• Definition: "Concealing information and resources"
• E.g., access control to documents/features
• Confidentiality applied to RELAI?
• The system shall ensure that private conversations are not shared with anyone.
• The system shall provide mechanisms for patients to exercise their "right to forget".
• The system shall not exchange/store data that were not anonymized.
• The system shall only store patients' messages, not answers

Integrity
• Definition: "preventing improper change so that you can trust the system"
• E.g., with authentication and detection mechanisms
• Integrity applied to RELAI?
• The system shall ensure that suicidal risk evaluations are done on the right data.
• The system shall guarantee that the anonymization method does not interfere with the provided assistance.
• The system shall guarantee that data impacted by a "right to forget" request are removed from the NLP model in a correct way.

Availability
• Definition: "are resources and data actually usable?"
• E.g., Prevention of Denial of Service (DoS) attacks.
• Availability applied to RELAI?
• The system shall collect the textual conversations in less than 5 minutes (time to fill out admission paperwork)
• The system shall provide its suicidal risk evaluation in less than 10 minutes (time between nurse appointment and psychiatrist one)
• The front-line mobile app shall be available at 99% (7.5h downtime per month)
• The front-line mobile app shall redirect to a phone-based service if unavailable

Scenario: Arriving at the Psychiatric ER
1. Patient enters the Psychiatric ER service
2. Patient is evaluated by Nurse, who offers to use the RELAI app if relevant
(a) If Patient does not accept, proceed as usual (Provide Care - Regular)
3. Patient signs the consent form
4. Patient install the RELAI Mobile app (RMa) on their phone
5. RMa collects eight weeks of textual conversations on the phone
6. Patient fills out a form to describe their relation with the identified persons
7. RMa send the conversations and form to the NLP model
8. The NLP analysis (e.g., suicidal risk evaluation) is sent to Practician
9. Practician proceeds (Provide Care - RELAI augmented)

Risk Management

What can go wrong? 1. The ethics committee(s) does not approve the project **2**.Personal messages leak on the internet **3**.No patients agree to participate in the study

• **How to mitigate such risks? 1.** Assurance process to support ethics approval **2**. Anonymization and encryption for messages, authentication between app and servers. **3**. Use publicly available datasets for training and reduce the ambition of evaluation.

Constraint: Ethic Committee(s)

• The project operates under three jurisdictions: Canada, Belgium, Switzerland
• We need an assurance process to convince ethics committees to let us: Collect the data, Train the models, Operate the app

Storing the dataset

• **Situation**: We're collecting messages from a patient in Switzerland or in Belgium. The training dataset is hosted at CIRST in Montreal

• **Can we specify some requirements here? 1.** The system must anonymize the conversations before sending them to external
systems **2**. The system must ensure secure communication between the patient's phone and the dataset storage **3**. The system must store the dataset in a way that prevents its usage by non-authorized personnel (only NLP researchers).

Anonymizing text messages

• **Situation**: input: "Hey Alex, I'm not feeling well, can we have a coffee on Locke? " Output: "Hey <Part76>, I'm not feeling well, can we have a coffee on <Loc12>? "

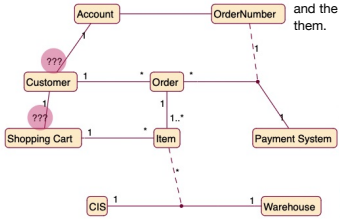
• **Can we specify some requirements here? 1.** The anonymization must preserve participants' semantic links **2**. The anonymization must guarantee the uniqueness of <placeholders> **3**. The anonymization must guarantee that enough data are anonymized so that it is not possible to identify the authors.

Right to forget (RTF)

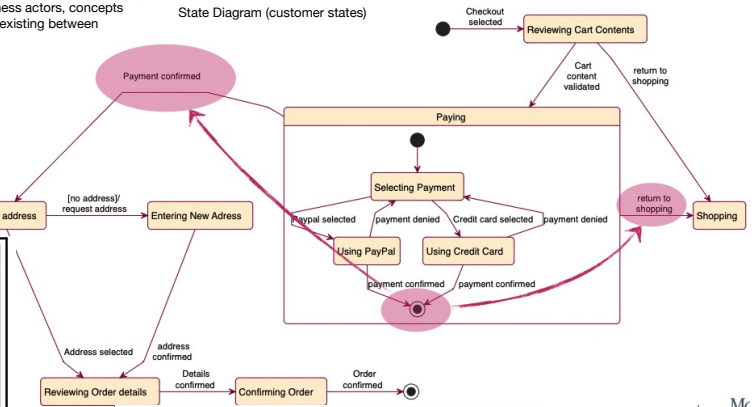
• **Situation**: Avery consented to participate in the RELAI training dataset. Avery has now changed their mind, and wants their message removed.

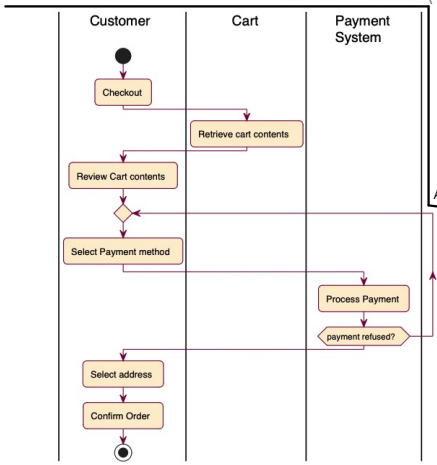
• **Can we specify some requirements here? 1.** The system shall identify the origin of messages in the training data set **2**. The system shall offer to patients a way to trigger an RTF request directly from
the mobile app, in less than 2 minutes **3**. The system shall restart a model training phase and re-deploy the resulting models after having removed messages covered by an RTF request.

CONGO CASE STUDY: Domain model —> Looking for Business actors, concepts and the relations existing between them.



State Diagram (customer states)





Activity Diagram