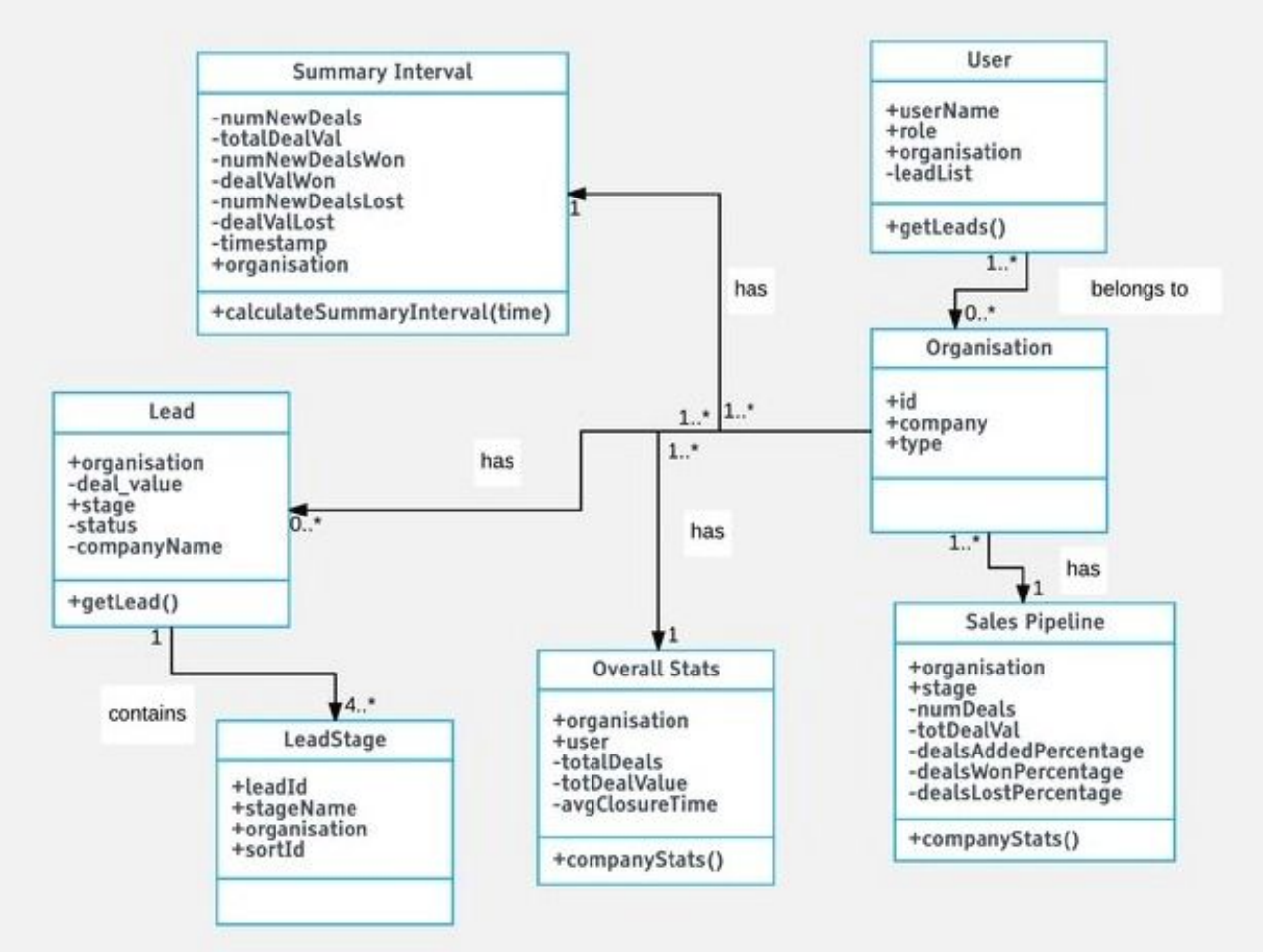
Product Design

TEAM

TEAM NUMBER	**SSAD47**
TEAM NAME	**Sales Assistant Bot**
MEMBERS	**1. Mugdha Yashodhan Abhyankar** **2. Buggana Sathvik Sanjeev** **3. Koushik Sai Achyuth Ayila**

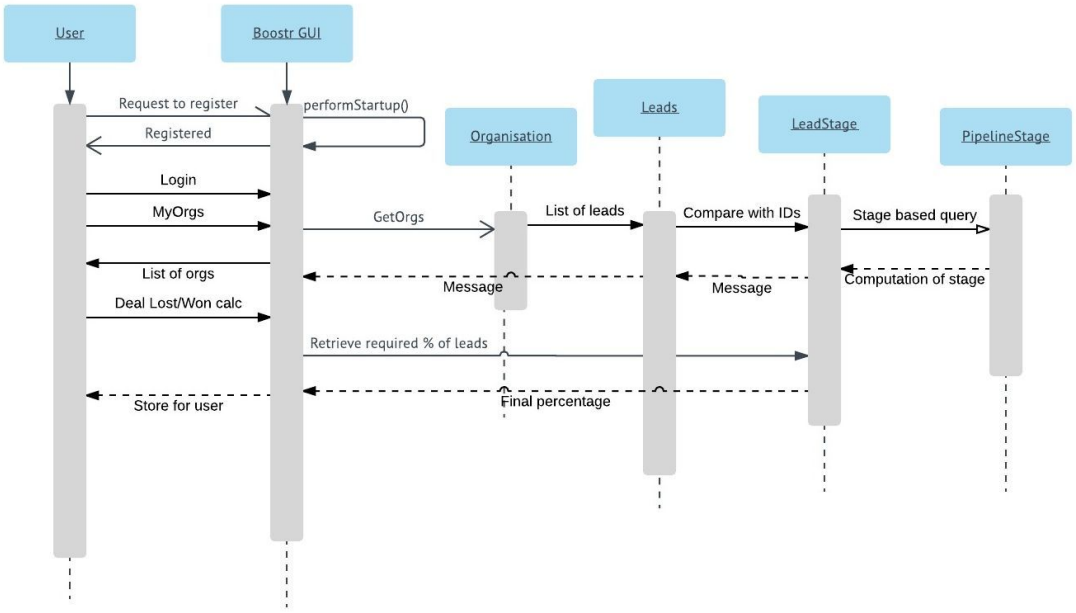
Design Model



Summary Interval	Class state ● The class contains number and value of the deals added, won and lost on a monthly, weekly and daily basis. Class behavior ● The class contains a method to compute the above on the basis of duration requested.
Overall Stats	Class state ● The class contains the deals statistics and average time of closure of leads on a company level Class behavior ● The above data can be retrieved through a method
Sales Pipeline	Class state ● The class contains the deal statistics for every stage of the lead that belongs to the organisation Class behavior ● The above data can be retrieved through a method
Lead	Class state ● A lead has the organisation it is defined for, the stage that it is in, its deal value and the name of the lead (company) Class behavior ● A method gets all the above attributes.
User	Class state ● A user has a name, role, organisation, and a list of leads Class behavior ● The leads of a user can be extracted using a method
Organisation	Class state ● An organisation has an id, an associated company, and the company type (vendor / partner, etc)
Lead Stage	Class state ● This class stores stage related information of every lead

.

Sequence Diagram(s)



Design Rationale

Every user has a list of organisations he/she is attached to. These organisations have a set of leads it is working on. Every lead has a stage data to maintain, and the information associated with deals. The best way to design this therefore is to have different class performing individual tasks, and integrating them. This ensures loose coupling.

The Summary Interval, Overall Stats and Pipeline Stats are the computations and analysis made on the leads present. They therefore extract data from the lead related classes and does its calculations. The user and organisation are black boxes to these classes.

This approach has a lot of benefits like loose coupling within the different users, proper division of responsibility and good modularity. The calculation classes however have a considerable dependence on the basic modules.