Advanced Topics in Software Engineering Project

**CRM System**

Team Name:

AC/DC

Team members:

1. Mohsen Shafiey Kasrineh
2. Shayan Doroodian
3. Pascal Atosha
4. Farshid Cheraghchian

Project Descriptions

**CRM FEATURES:**

1. Workflow automation

CRM software can make your life easier by automating your workflow. Look for software that lets you set up custom rules and comes with sales force automation to help you and your team be more productive.

1. Customization

No two businesses are the same, so the best CRM software can cater to your business’s unique needs. Some CRM solutions let you choose which features to include in your package, and some also allow you to customize basic areas, such as with the abilities to add contact fields, choose which data to show on your dashboard and create custom reports. You can also customize with extensions, plugins and other add-ons to expand your software’s capabilities.

1. Third-party integration

Connecting your CRM software to other solutions you already use can save you tons of time and money. Most CRM programs offer third-party software integrations; the key is choosing one that’s compatible with the software you use and easy to implement.

1. Customer service

CRM software can help you acquire and retain customers by providing excellent customer service.

1. Employee tracking

CRM software is a great way to track employee activity and performance. Choose software that gives employees their own accounts where they can individually track their hours, tasks, meetings, sales numbers, goals and other items.

1. Social media

A CRM with built-in social media monitoring features can save your marketing team time, because they won’t have to switch platforms to perform the essential task of monitoring your company’s social media.

1. Lead management

A lead management feature will help you to identify your leads and the actions they’ve taken along the sales cycle. Through the lead management process, you’ll be able to score your leads and, if needed, filter them off to a different member of your team to turn select leads into customers.

1. Real-time data

Starting in 2017, real-time data became the main driver for CRM systems, according to CMSWire. As noted in the article, CRMs can pull real-time data from devices, applications and even appliances.

1. CRM analytics

CRM analytics, also known as customer analytics, offer insight into customer data. This data is important because it can help you make better decisions about the types of products, services, marketing and overall communications you distribute to your customers.

1. Reporting

Reporting is the feature that brings the results of your sales and marketing efforts all together, and it comes standard with all the CRMs we’ve researched.

1. Sales forecasting

A CRM with this feature predicts your future sales based on data from your current pipeline. You may need to alter your marketing plan based on the sales forecast. For example, if you thought your toy store was going to sell a lot of girls’ dolls during Christmas but learned through sales forecasting that building blocks are more popular, you can cater your campaigns accordingly. Sales forecasting can also anticipate market changes to mitigate your business risks.

1. Email

Tracking emails through a traditional inbox can get overwhelming and confusing. Livewire noted that, as of 2015, the average office worker received 121 emails. It’s important for every single sales email to be received and addressed in a timely fashion to increase the probability of generating revenue.

**Extreme programming:** Extreme Programming (XP) is an agile software development framework that aims to produce higher quality software, and higher quality of life for the development team. XP is the most specific of the agile frameworks regarding appropriate engineering practices for software development.

**User Stories:** Describe what the product should do in terms meaningful to customers and users. These stories are intended to be short descriptions of things users want to be able to do with the product that can be used for planning and serve as reminders for more detailed conversations when the team gets around to realizing that particular story. User Stories are written by the customers as things that the system needs to do for them. They are similar to usage scenarios, except that they are not limited to describing a user interface. They are in the format of about three sentences of text written by the customer in the customer's terminology without techno-syntax.

In our case, the system will have three permission levels. The “employee” Level, the

“Manager” level and the “Director” Level. Each level will have different

permissions. Also, the customer must be able to select from a drop-down box

if he wants to have French/English/Farsi Interface (Multilanguage support). The

employee will be able to have access only to Manage Calls, the Manager will

have access to Manage Customer. The director will have access to all sections

including Manage Reports.

The system will have three sections in the menu.

a.  Manage Customers – A list of customers using a grid control where he/she

can add, edit or delete a customer. The grid control must support

paging of 10 customers per page.

b.  Manage Calls of each customer. – A list of all calls made by the

customer and be able to add, edit or delete a specific call. The grid

control must support paging of 10 calls per page filtered by customer.

c.  Manage Reports. The system will prepare 1 report. The report will

have a list of customers with their calls per customer within the same report.

**assumptions:**

* There are no stakeholders in this project
* Team contains four developers
* Users are imaginary

**Developer**

as a developer:

* I want to be in touch with all the crew everyday

So, I can suggest my opinions and get feedbacks

* I want to use an agile method

So, I can feel being a part of project owner, not an employee

* I want to learn new platforms and technologies from other developers

So, I can improve my performance and help others to improve theirs

* I want to be sure other developers meet their tasks on time

So, I can resume my tasks with minimum stress

* I want to receive short term and long-term feedbacks from users

So, I can have a good background to improve and maintain the project

* I want to have all the permissions in GitHub

So, I can push my changes whenever it ends

**Customer**

1- as a director

* I want to have permission to delete managers and employees and technical supporters' account

So, I can prevent their access, after they leave the company

* I want to record all the conversation between employees and customers and technical supporters and customers

So, I can control and improve the quality of services

* I want to meet all the team online or face to face every weekend

So, I can communicate with them and find out the strength and weakens of them and help them to enhance their job functionality and their self-steam

* I want to access to all reports and Data

So, I can monitor them

2- as a manager

* I want to have permission to delete customers' account

So, I can prevent their access, after they leave the software

* I want to have a fast and easy to use database

So, I can put the data with most comfort and less complexity

3- as an employee

* I want to have a fast and easy to use database

So, I can put the data with most comfort and less complexity

* I want to have an easy to use customer calls environment

So, I can monitor customer calls and manage them without complexity

4- as a technical supporter

* I want to have permission to access to customer accounts

So, I can help and guide them with most effort

* I want to record all calls without deleting permission

So, I can listen to them and improve my dialogues

5 - as a customer

* I want to know what data are recorded

So, I can be sure that my personal information will not record

* I want to access an easy-to-use environment

So, I can put my calls without any complexity and waste of time

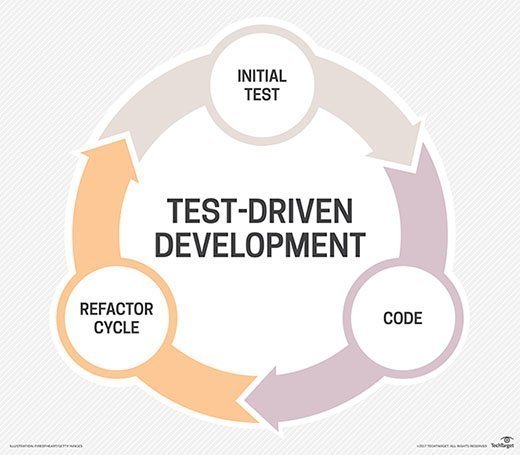
* I want to use fingerprint login

So, I can be sure nobody can access my profile but me

**Requirements Engineering:** Extreme Programming **performs requirements engineering throughout the lifecycle in small informal stages**. The customer joins the development team full time to write user stories, develop system acceptance tests, set priorities, and answer questions about the requirements.

**Test Driven Development:** In the industry, this cycle is referred as "red-green-refactor," named after the colors unit test frameworks traditionally use to express failure or success when running unit tests, TDD and unit-testing run together and complete each other.

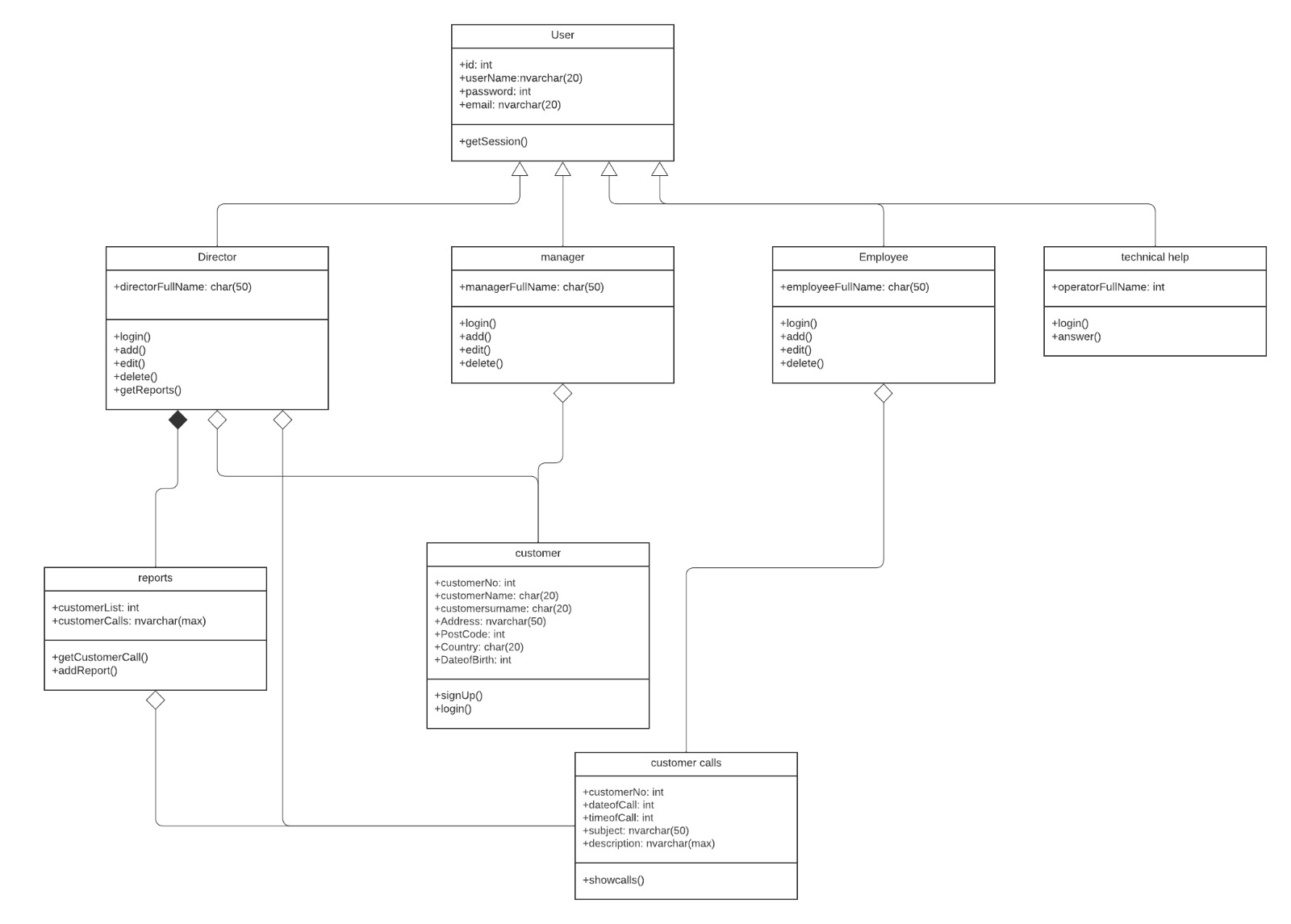
**Test Driven Development Cycle**



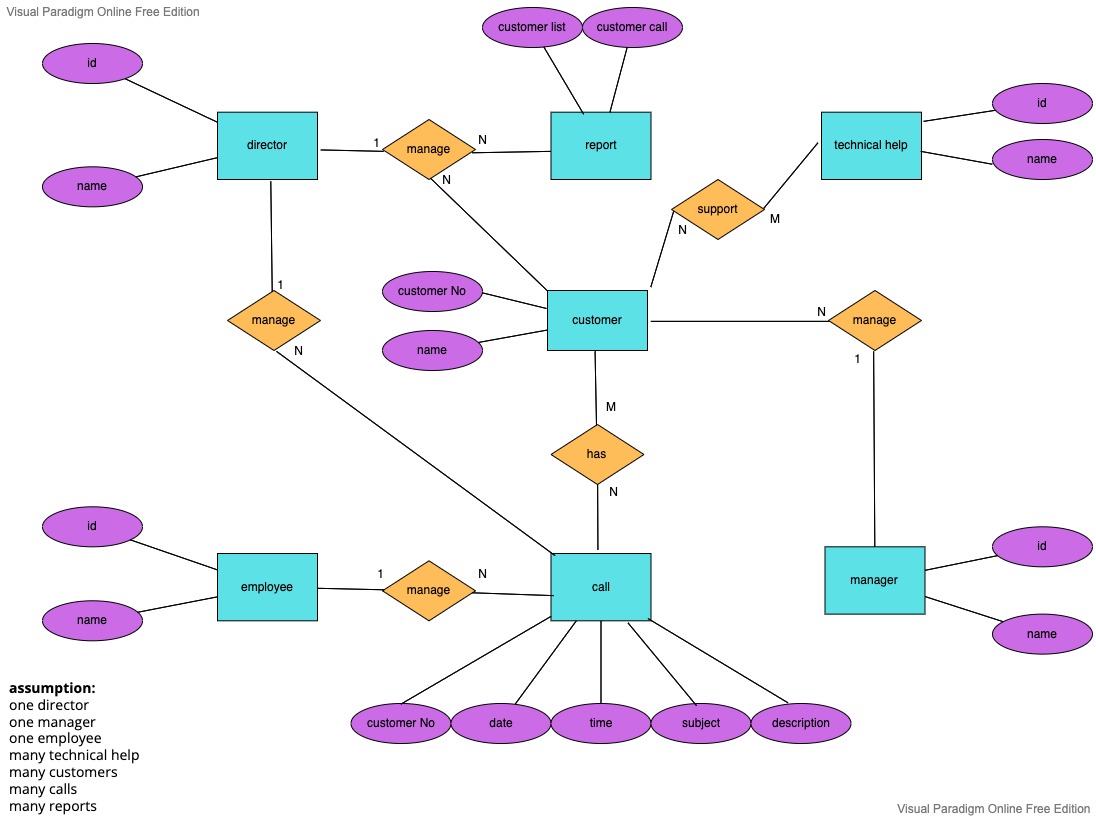
Project Tools

* In this project we are going to use "Kanban" as our management technique.
* For the data base we are using "Django" technology beside "python" for backend.
* Project interface is created with "HTML5" and "CSS3".
* We arranged daily meeting with "google meet" to check and review the whole process.
* We will add and change various tasks on the "GitHub Kanban board" to have a structured workflow.
* We are using "GitHub" as our version controller.
* We created our diagrams with "lucid.app" and "online.visual-paradigm.com"

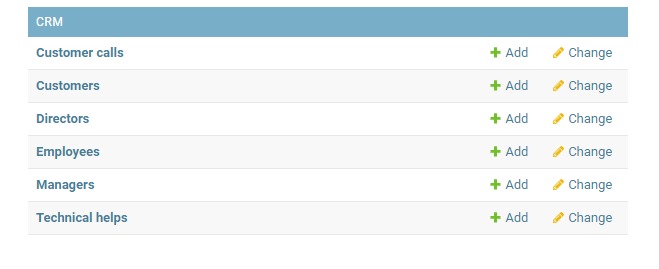
Class diagram for our CRM:



Relation diagram for our CRM:



Creation of models:



Our Meetings

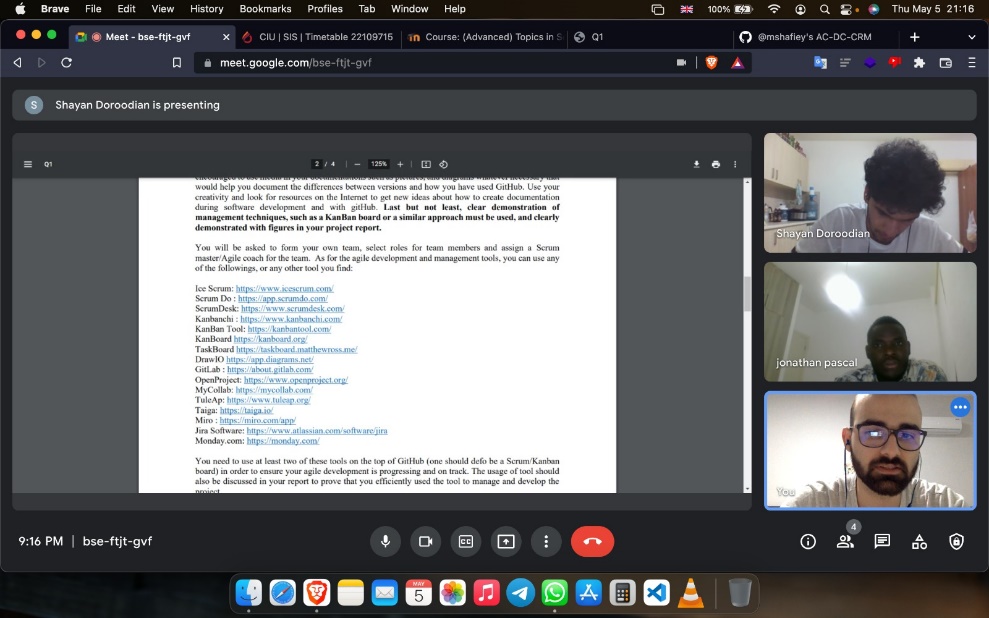
1. Apr17 – was our first meeting and we discussed about the whole project and send email to submit it.
2. Apr28 – we were talking about the concept of project and technology that members have clue in. and also customer needs and project specifications



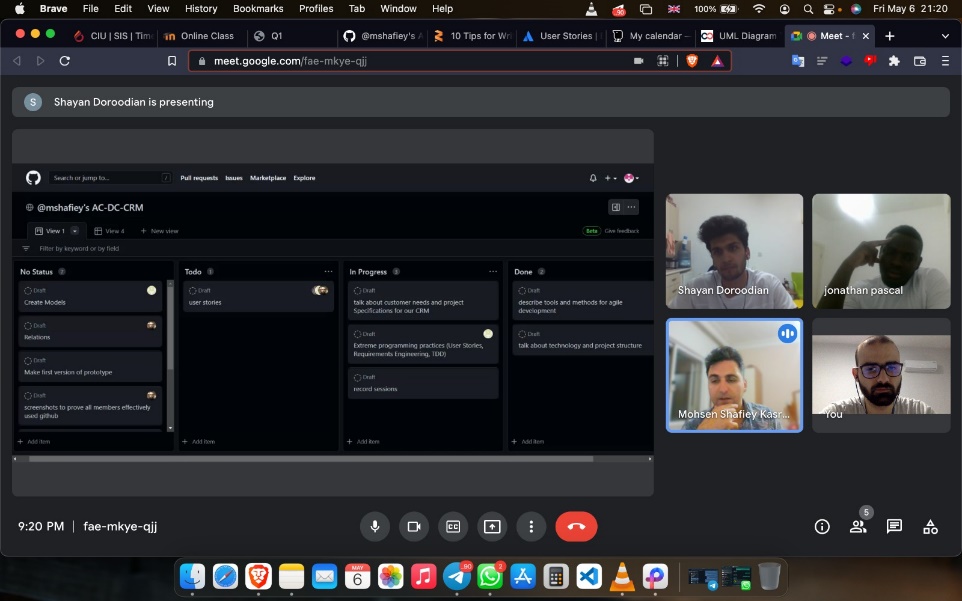
1. Apr29 – define initial tasks and assign it to every member on GitHub Kanban board, talk about technology and project structure



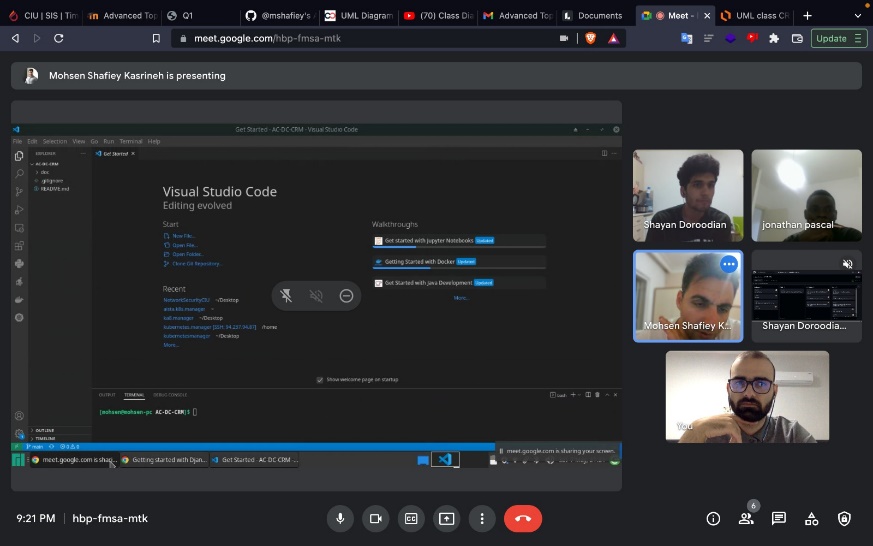
1. May5 – we check again the requirements pdf and update tasks and define the agile development that we are going to use. furthermore, we divided the whole process into 3 versions and put some milestones and make class diagram.



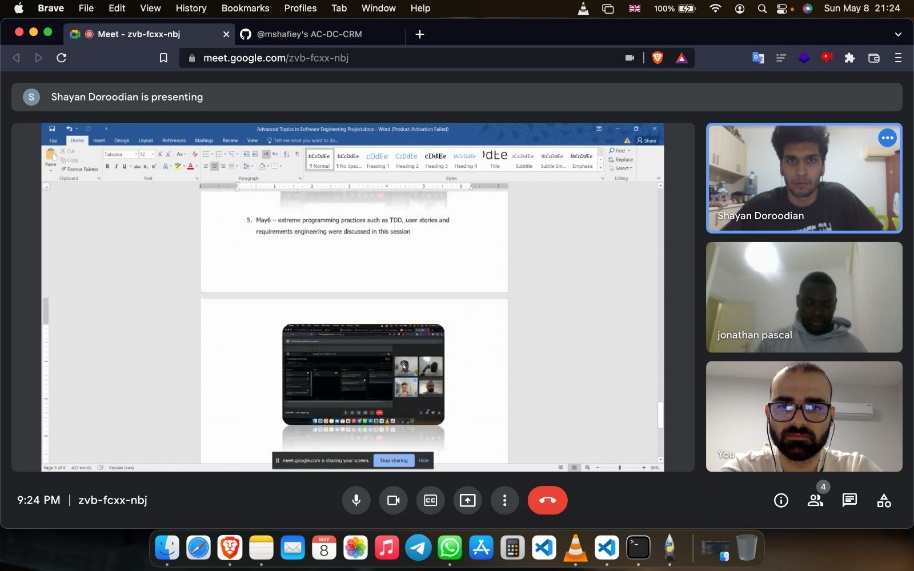
1. May6 – extreme programming practices such as TDD, user stories and requirements engineering were discussed in this session



1. May7 – we started to create our first version of software



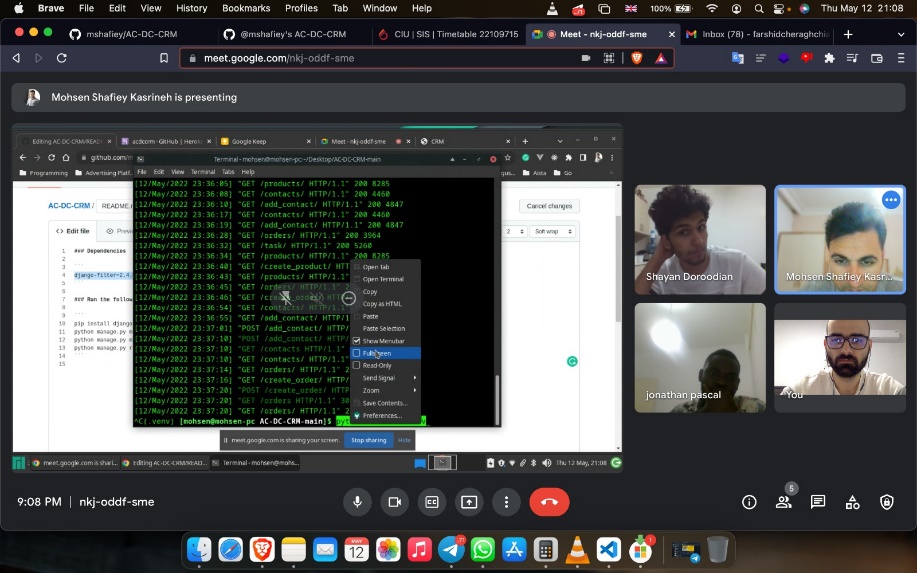
1. May8 – we finished and lunched our first version of software with doing pair-programming.



1. May9 – we discussed about new requirements and user stories two improve and reach at the second milestone.

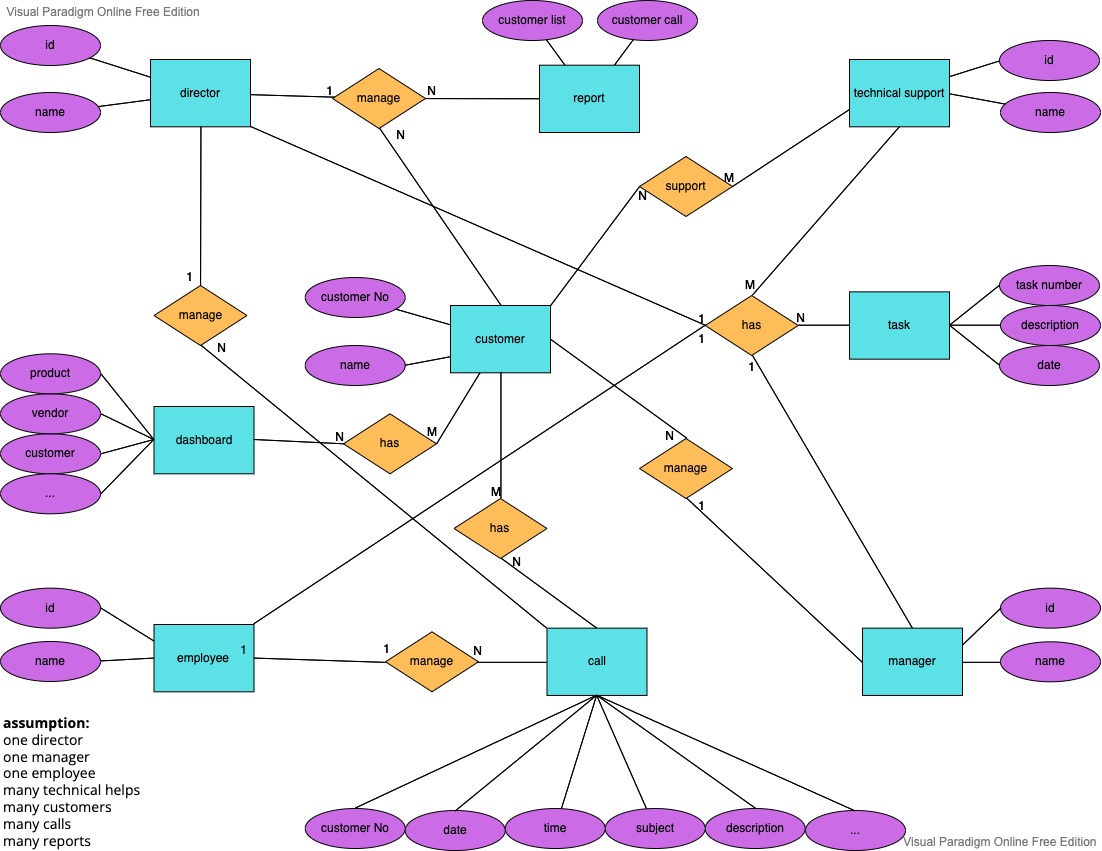
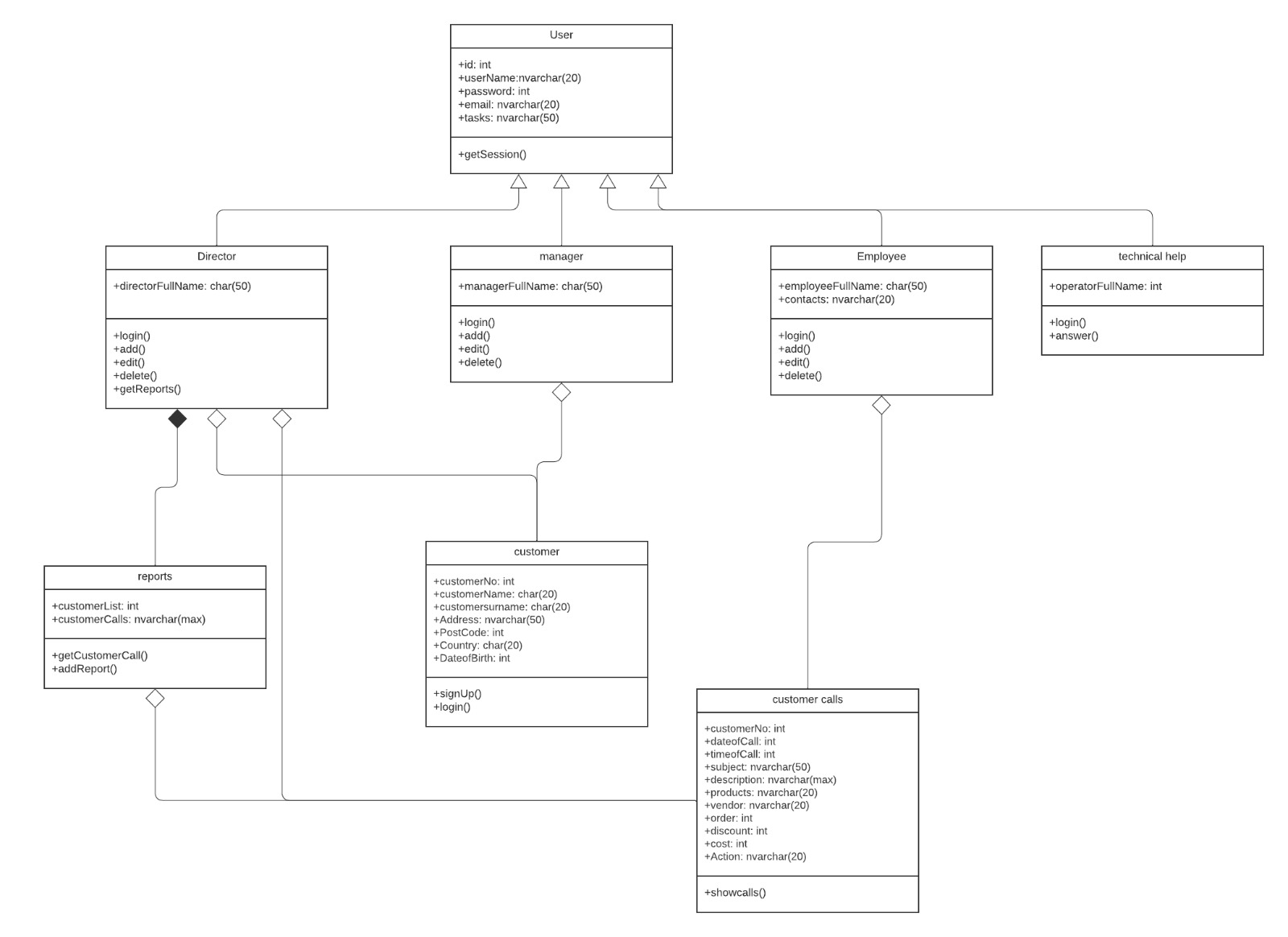


1. May 12 – we added some new features based on our decisions to improve system and then release new version to reach second milestone



Moreover, we use "pair programming" technique here, such that, Front-End development was assigned to "Shayan Doroodian" and "Farshid Cheraghchian", and Back-End development was assigned to "Mohsen Shafiey Kasrineh" and "Pascal Atosha" as well, to divide new tasks and workflow among all members.

In this level we had to implement new diagrams due to the presence of new features so we created these new ER and class diagram:



1. We tried to publish our system on "heroku" but admittedly we faced to some security issues due to new policies according to this source: (<https://github.blog/2022-04-15-security-alert-stolen-oauth-user-tokens/>)

