Bilkent University

Department of Computer Engineering

CS 353 Term Project

2018-2019 Spring Semester

Group 21

National Judiciary Informatics System

Project Proposal

Project Group Members

	Name	Section No.	ID
1.	Doğa Oruç	Section 2	21602022
2.	Faruk Oruç	Section 2	21601844
3.	Mert Soydinç	Section 3	21601321
4.	Ünsal Öztürk	Section 3	21601565

Supervisors: Özgür Ulusoy, Uğur Güdükbay

Teaching Assistant: Arif Usta

Table of Contents

Contents

1 Introduction

For the 2018-2019 Spring Semester CS353 Database Systems Term Project, Group 21 was assigned to develop a "small fraction" of a "National Judiciary Informatics System", heavily influenced by and based on the Turkish national judiciary informatics system, UYAP. The system will be called NJIS2.

This document contains information regarding the problem statement for the application to be proposed, project requirements, domain analysis, entities, actors, overview of high-level subsystems involved in the application, relationships between entities and actors, limitations of the software system to be proposed, and how a relational database management system plays a central role in the solution domain.

2 Proposed System

2.1 Problem Statement

Since the rise of SQL in the relational era of the 1980s, it has been commonplace for several private and public institutions to transfer tabular physical data (e.g. typewritten documents, hand-writing) to the digital domain for easier accessibility to and querying of the data.

As with most other institutions, due to the growing size of data and the demand for remote, reliable, and quick access to this data, it was seen appropriate to implement a software system based on monolithic relational database management systems in the judicial domain. It can easily be argued that such a system is far superior in terms of making data accessible and queryable in comparison to keeping tables in physical ledgers. This makes it such that whenever an actor (e.g. judges, lawyers, suspects) wishes to access or query data related to a particular court case, the actor does not have to go through stacks of paper to find the related information; the actor just has to ask the judiciary informatics system for the digital retrieval/materialization of this information, and could even do so from a remote location.

This facilitation of information transmission therefore forms the problem statement for this project: It is expensive, inefficient, and in some cases physically demanding to keep track of data related to judicial cases on paper; and thus it is very desirable to implement a standardized software system for the storage, querying, and the manipulation of the data related to the aforementioned judicial cases, based on a RDBMS.

2.2 Overview

We will implement a web-based judicial informatics system inspired by UYAP (Ulusal Yargı Ağı Bilişim Sistemi) using Java and Java Server Pages based on Oracle as its RDBMS. The reasoning behind choosing these systems for our implementation is that we perceived these frameworks to be rather popular in the industry, and will allow other engineers to easily understand the software system in the case that it is required to extend/adapt this system to other judicial domains or even entirely different domains.

The main properties of our proposed software system are as follows:

- There are four types of users, each corresponding to a role in the judiciary system. These roles are: Judges, Lawyers, Citizens, and Expert Witnesses. Each type of user will have a login with different set of functionalities and permissions based on their judicial role.
- Authenticated lawyers will be able to start court cases in the name of a particular citizen. This communication may or may not be through the proposed system; however, this information will be stored.
- Authenticated citizens may view their judicial status, i.e. which court cases they are part of as witnesses, victims, or suspects. They are allowed to view this status but they are not allowed to edit it.
- Authenticated judicial staff may view the court cases in which they are involved, and they have access to all information related to the court case, including evidence, lawyers involved in the case, eye witnesses, expert witnesses, prosecutors, suspects, victims, case interpreters, guardians (for minors), judges.
- Authenticated conciliators may also access the system in order to handle minor court cases without any involvement of the judiciary staff, such as employer-employee debt cases, and job-related demand cases.
- The system will store information regarding the state of court cases. The state of the court case may be viewed, updated, and closed if deemed necessary by a subset of the authenticated judiciary staff or the system administrator.
- The related information about all actors, court cases, login information, actor to role mapping, actor to court case mapping will be stored in a RDBMS.

2.3 Necessity of Using a Relational Database Management System

RDBMS is a necessity in the implementation of the system because of the functionality it provides in terms of the transactional integrity when it comes to creating, reading, updating, deleting data related to these judiciary cases, as it is not mentally demanding to perform non-anomalous transactions on the data stored in at least the third normal form. Transactional integrity is crucial for this domain, since faults and bugs in the system may not be tolerated as they would cause dire consequences for the parties involved.

Another reason for using a RDBMS is that it is very suitable to model the system in terms of entities and relationships between these entities, and RDBMSs are inherently based on the principals of relational algebra and its extensions. Also, on a very low-level, monolithic RDBMSs provide low-latency access/querying to/of the data, typically in the order of microseconds, given that the system is not storing huge amounts of data, i.e. data that exceeds the current maximum storage for a single machine.

Finally, using a database management system also allows the system to be adaptable to changes in laws, or in ways in which certain court cases are handled, since developers do not have to change much in the back-end implementation of the system. New tables, relations, and properties may be added in the database system much more freely, given that the developers are able to maintain 3NF.

3 Requirements

3.1 Functional Requirements

3.1.1 Actors and Their Functions

The following are the definitions of the actors that are to be supported by the judiciary system and the actions they may take in the system.

- Litigant: Actors involved in court cases with the following permissions:
    - May view court cases that involve them.
    - May view the current state of the court case.
    - May view the people involved in the case.
    - May not start court cases.
    - May not create, read, update, or delete files.
    - May call guardians or eye witnesses for the court case for their side.
    - Called to court as a litigant for the court case.

- Eye Witness:
    - May view court cases that involve them.
    - May view the current state of the court case.
    - May view the people involved in the case.
    - May not start court cases.
    - May not create, update, or delete files.
    - May read files.
    - Called to court as an eye witness to the court case.

- Guardian:
    - May view the court cases that involve them.
    - May view the current state of the court case.
    - May view the people involved in the case.
    - May not start court cases.
    - May not create, update or delete files.
    - May read files.
    - May view the people involved in the case.
    - Called to court as a guardian for a minor for the court case.

- Judges: Actors involved in court cases with the following permissions:
    - May view court cases that involve them.
    - May view the current state of the court case.
    - May view the people involved in the case.
    - May not create court cases.
    - May read files related to the case.
    - May not create, update, or delete files.
    - Receive pending court cases assigned to them by the system.
    - May change the status of a court case from open to closed.
    - May not change the status of a court case from pending to open.
    - May cast a verdict for a court case to close it.
    - May not request to create court cases.

- Lawyers: Actors involved in court cases with the following permissions:
    - May view court cases that involve them.
    - May view the current state of the court case.
    - May view the people involved in the case.
    - May request to create a court case for certain cases.
    - May create, upload, and read files.
    - May not delete files.
    - Called to court as a lawyer to the court case.

- Public Prosecutor:
    - May view court cases that involve them.
    - May view the current state of the court case.
    - May view the people involved in the case.
    - May request to create a court case for certain cases.
    - May create, upload, and read files.
    - May not delete files.
    - Called to court as a public prosecutor to the court case.
- Interpreter:
    - May view court cases that involve them.
    - May view the current state of the court case.
    - May view the people involved in the case.
    - May not create court cases.
    - May create, upload and read files to provide translations.
    - May not delete files.
    - Called to court as a translator to the court case, if applicable.

- Conciliator:
    - May view court cases that involve them.
    - May view the current state of the court case.
    - May view the people involved in the case.
    - May not create court cases.
    - May not create, update, or delete files.
    - May read files.
    - Can arrange a meeting between the litigants outside the court.

- May inform the judge about the outcome of the meeting.
- May be called to the court as a conciliator.

3.1.2 Interactions between Actors and the Life-Cycle of a Case

The judiciary system models the dynamics between the actors as follows. For a regular court case, citizens who are registered in the system may request from their lawyers to file a request for opening a court case to the system. Then, the court case is marked as pending and the system allocates an available judge to the case. Once the judge is notified that the judge is assigned a case, the state of the case is marked as open. Whenever the initialization of a court case is finalized, citizens who are involved in the case are now litigants. At any point before the scheduled day of the court, the defending litigant may notify a lawyer to participate in the court case for the defending side. The participants of the court case may perform their allowed actions before the judge closes the case by casting a verdict.

Cases which involve manslaughter, homicide, or any other type of case in which the citizen may not legally ask a lawyer to file a request for opening a court case, a public prosecutor may, independent of the will of the citizen, file a request for a court case. The rest of the process is the same as a regular court case.

There may be conciliators assigned to an open case if the case is deemed to be too “trivial” or “small”. In this case, the conciliator is to arrange a meeting between the litigants and attempt to solve the problem without further involvement of lawyers, judges, and prosecutors via bargaining between two parties. If the two parties can come to an agreement, the conciliator is to inform the judge to close/finalize the case, as in this particular case, the parties reached an agreement. In the case that no agreement is reached, the conciliator is dismissed, and the rest of the process is the same as a regular court case.

There may be extra parties involved in a court case. Those parties are eye witnesses, guardians, and interpreters. Eye witnesses are to provide extra evidence, guardians are to be present in a court case if at least one of the litigants is a minor, and interpreters are present when translation is necessary for the court case. Eye witnesses and guardians may be assigned to the court case by litigants, in the case that they are available/required. For interpreters, the system will assign interpreters to litigants of other nationalities. After the first court hearing, the litigants may dismiss their interpreters if they deem it to be necessary.

3.1.3 Performable Actions in the System

- Request to open a court case: The appropriate actor may perform this action to request from the system to generate a new court case. The system then completes this request.
- View current state of case: A case may be pending, open or closed. Any actor may see the current state of the case.
- Implicitly set state of case: A case may change its state upon reaching certain milestones in the life-cycle of a case. The case is pending when it is initially created. The case is open when a judge is assigned to the case. The case is closed when a judge explicitly closes the case. These are the only conditions under which the current state of the case is changed.
- View people related to the case: A list of people involved in the case are provided to the user.
- Create, read, update, delete: An authenticated user may create, read, update, or delete files related to the case. No actor is allowed to delete a given file to prevent corrupt behavior of actors.
- View court cases: Any actor can see the court cases that they are involved in.
- Messaging: Any actor can send a message to another actor.
- Arrange meeting: Conciliators may arrange meetings between the parties.
- Login: To determine the role of a user, the user has to login with a set of credentials. Upon a successful login, the user will have access to the system with their particular role.

3.2 Non-Functional Requirements

3.2.1 Usability and User Interface

The user interface for the judiciary system should be simple to use and understand, while maintaining functionality and efficiency in the representation of the data. The users should be able to locate the actions they may take relatively easily in the user interface, and should be able to learn how to navigate through the system with very high efficiency. This can be achieved via clever UI design.

3.2.2 Reliability and Maintenance

The system should be reliable in the sense that queries to the database are not erroneous syntactically or logically, so that results are always correctly returned. The information that should be hidden from certain actors should be carefully implemented. The UI should not crash because of problematic components. The transactions to the database must not introduce anomalies to the data. The data must be stored in at least the third normal form. The tables should be cleverly designed for future monolithic scalability in terms of transactional integrity and simple maintenance.

3.2.3 Performance Concerns and Constraints

User experience must satisfy certain criteria, via service level agreements, and for this reason, the system has to perform up to a standard. The following has to be satisfied:

- ACID Constraints: The system should maintain atomicity, consistency, isolation, and durability.

- Capacity: The system should store and retrieve data efficiently. If the data stored is too much for the hardware to handle, scaling up is necessary. If it is no longer possible to store data on a single machine, the system has to be redesigned in the domain of distributed systems and NoSQL. Hence to keep RDBMS transactional integrity, the system should be designed in such a way that the problems caused by the growing amount of data can be reduced by simply scaling up in hardware.

- Latency: The system should have a latency in the order of microseconds in terms of query execution. If it is necessary to keep the latency low for queries, the hardware should be scaled up. If it is no longer to scale up the hardware, popular queries should be precomputed and cached in memory. If a value for a query was changed, the precomputed query must be re-computed to ensure consistency. If pre-computations become an overhead, this system has to be redesigned in the domain of distributed systems and NoSQL. Hence to keep RDBMS transactional integrity, the system should be designed in such a way that the overhead caused by the growing amount of queries can be reduced by simply scaling up in hardware.

3.3 Pseudo Requirements

The pseudo requirements for the system are as follows:

- For the relational database management system, Oracle will be used.
- For the development technology/back-end development, JSP will be used.
- For the programming language, Java will be used.
- For web technologies, HTML, CSS, JavaScript will be used.

4 Limitations

The following limitations have been put into place so that the underlying systems of the judicial system are not abused via spamming data or spamming invites to court cases.

- Judges and lawyers may not be assigned to more than a predetermined amount of cases, by default 5, simultaneously.

- No actor may delete files from the system to prevent corruption in the court case.

- Lawyers and prosecutors have a daily limit of requesting to open court cases in the system, determined by a predetermined value, by default 2.

- Guardians, interpreters, conciliators, expert witnesses, and eye witnesses, may not be assigned to more than a predetermined amount simultaneously.

- Files larger than a certain amount may not be uploaded to the filesystem, the limit being 128MB by default.

- The number of files that a certain user may upload to a court case has a daily limit of a predetermined amount, by default 3.

- The length of a message that can be sent to another user has a maximum predetermined character limit, by default 255.

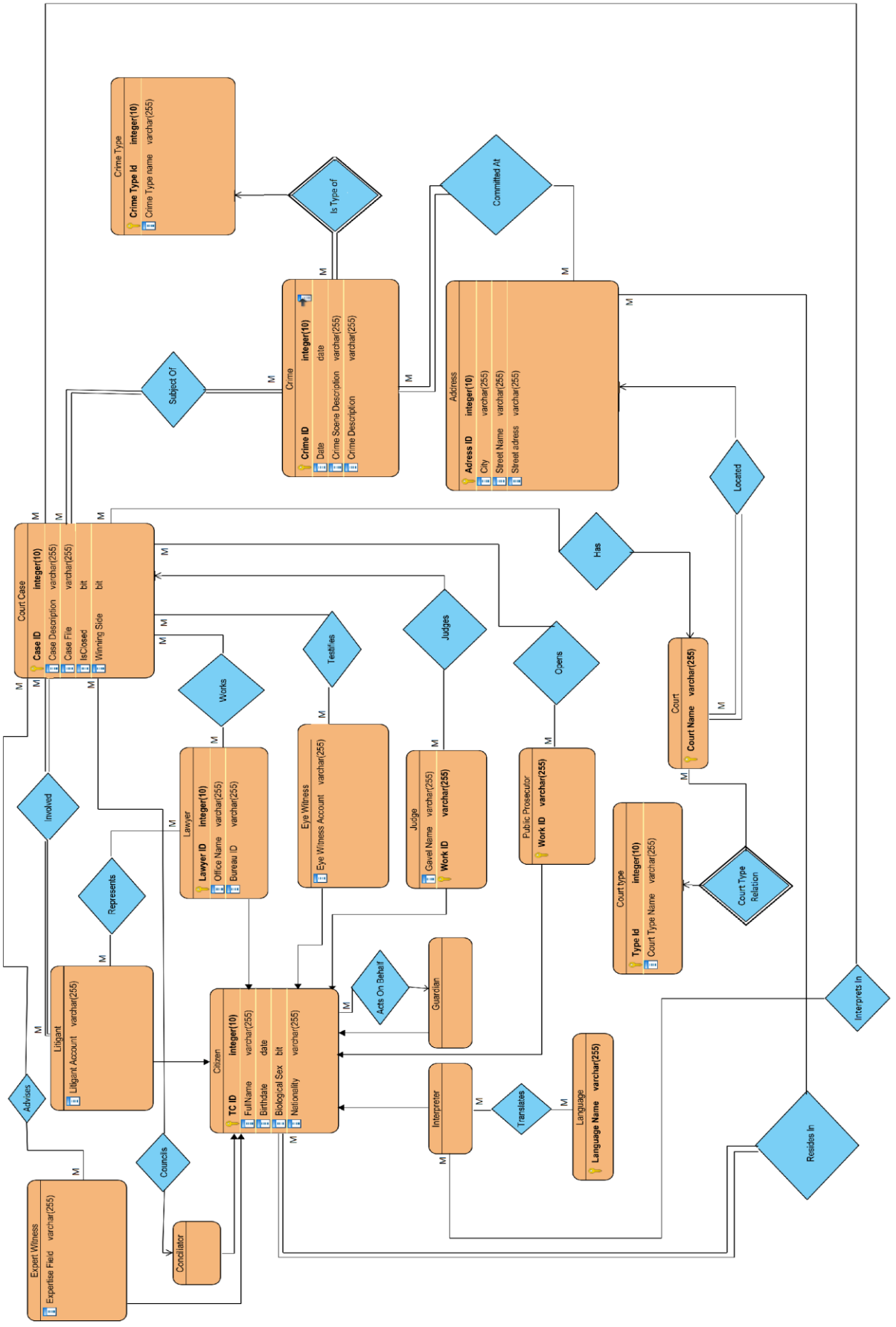
The following are the inherent limitations of the judiciary system.

- The system does not handle the court cases in which litigants are corporations, government institutions, or any other *group of people*.

- Actors have to have logins, and their type of login determines their role and permissions in the system. This raises some security concerns. The passwords/login credentials have to be stored in the database in hashed form, which increases latency a little bit, since during login, one has to hash the entered password to be compared to the stored hash version in the database.

- Hardware limitations: It is not possible to scale up beyond a certain limit, and in the case that one has to scale up beyond contemporary technological capabilities, the system must be redesigned in a distributed fashion, using NoSQL strategies, which would most probably result in the system having to sacrifice either consistency, availability or partition tolerance according to the CAP theorem, which is not desirable given the domain of the system.

5 E/R Diagram


Crime Type
Crime Type Id integer(10)
Crime Type name varchar(255)
Is Type of
Committed At
Crime
Crime ID integer(10)
Date date
Crime Scene Description varchar(255)
Crime Description varchar(255)
Address
Adress ID integer(10)
City varchar(255)
Street Name varchar(255)
Street adress varchar(255)
Subject Of
Located
Has
Court Case
Case ID integer(10)
Case Description varchar(255)
Case File varchar(255)
IsClosed bit
Winning Side bit
Works
Testifies
Judges
Opens
Court
Court Name varchar(255)
Involved
Lawyer
Lawyer ID integer(10)
Office Name varchar(255)
Bureau ID varchar(255)
Eye Witness
Eye Witness Account varchar(255)
Judge
Gavel Name varchar(255)
Work ID varchar(255)
Public Prosecutor
Work ID varchar(255)
Court type
Type Id integer(10)
Court Type Name varchar(255)
Court Type Relation
Represents
Litigant
Litigant Account varchar(255)
Acts On Behalf
Guardian
Citizen
TC ID integer(10)
FullName varchar(255)
Birthdate date
Biological Sex bit
Nationality varchar(255)
Interprets In
Advises
Interpreter
Translates
Language
Language Name varchar(255)
Councils
Resides In
Expert Witness
Expertise Field varchar(255)
Conciliator

5.1 Explanation for Entities and Relations

Entities:

- Citizen: Represents all users in the database. Has a unique TC ID, full name, birthday, biological sex, and nationality. Many other entities are inherited from this entity, namely: Interpreter, Guardian, Litigant, Judge, Lawyer, Public Prosecutor, Expert Witness, Eye Witness, and Conciliator.
- Interpreter: Translates between parties.
- Guardian: Assigned to minor litigants.
- Litigants: Derived from citizen. In addition to its fields, it has an account.
- Judge: Derived from citizen, in addition to its fields, it has a unique work ID and most importantly, *the name of the gavel of the judge is stored*.
- Lawyer: Stores unique lawyer and bureau ID, and has an office name.
- Public Prosecutor: Derived from citizen. In addition to its fields, it has a unique work ID.
- Expert Witness: Derived from citizen. In addition to its fields, it has an expertise field.
- Eye Witness: Derived from citizen. In addition to its fields, it has a path to the file for the eye witness account.
- Conciliator: Derived from citizen.
- Language: An entity that has a language name as a field.
- Court type: Enumerates type of courts. Has a unique type ID, and has a name for the type of court.
- Court: An entity representing a court. Has a unique court name.
- Court case: An entity representing a court case. Has a unique case ID, and has fields for case description, reference to the file of the case in the file system, and Booleans representing the winning side and the state of the case.
- Crime: Represents crime. Has a unique ID field, date, crime scene description, and crime description.
- Address: Defines an address. Has a unique address ID, stores the name of the city, street name and address.
- Crime Type: An entity enumerating the type of crimes. Has a unique type ID, and a name for the crime type.

Relations:

- Translates: A many-to-many relation between interpreters and the languages they are able to interpret. A language may be interpreted by many interpreters and an interpreter may translate multiple languages.
- Advises: A many-to-many relation between Expert Witnesses and the Court Case. Expert Witness provides his/her expertise on the events.
- Involved: A many-to-many relation between Litigant and the Court Case. Litigant participates in a court case as either a victim or a suspect.
- Represents: A many-to-many relation between Litigant and Lawyer. As the name suggests a lawyer represents a litigant.
- Councils: A many-to-1 relation between Judge and Court Case. The Conciliator tries to settle the case between the participants.
- Acts On Behalf: A many-to-1 relation between Citizen and Guardian. If the citizen is underage, a guardian is assigned to act on behalf of them.
- Is Type of: A many-to-1 relation between Crime and Crime Type. A crime must have a crime type (such as homicide).
- Testifies: A many-to-many relation between Eye Witnesses and the Court Case. Eye Witness provides his/her account of the events.
- Works: A many-to-many relation between Lawyer and Court Case. Up to 2 Lawyers per Court Case can be present.
- Subject Of: A many-to-many relation between Crime and Court Case. A single incidence of a crime can have more than 1 cases while a Case may include more than 1 crime.
- Judges: A many-to-1 relation between Judge and Court Case. While a case cannot have more than 1 judge, a judge can have many cases.
- Opens: A many-to-many relation between Public Prosecutor and Court Case. If a citizen is unable to open cases for themselves, Prosecutor opens the case for them or the government.
- Has: A one-to-many relation between Cases and their Court. A court case can have up to 1 court.
- Committed At: A many-to-many relation between Address and Crime. A crime could have more than 1 address and there could be more than one crime committed on the same address.
- Located: The address of the Court. In one address there could be several courts but a court could only have 1 address.

- Resides In: A one-to-many relation between Citizen and Address. A citizen must have at least 1 address.
- Court Type Relation: A many-to-1 relation between Court and Court Type. A Court must have a court type.
- Interprets In: A many-to-many relation between Interpreter and Court Case. There could be several interpreters in one case and they could work on different cases.

6 Project Webpage and Conclusion

The following are the links for the project and a higher quality version of the E/R Diagram.

Imgur Link: https://i.imgur.com/peuIj3r.png

Website Link: https://uensalo.github.io/Spring-2019-CS353-Group-21/

In this proposal, we provided the problem statement, our motivation to use a database management system to solve the aforementioned problem, along with the functional and non-functional requirements, the limitations, entities and relationships for the database component, and pseudo-requirements for the project. To conclude, this project will provide ease of access to judicial data through a web-based user interface that will be implemented with JSF and Oracle. We will try to conform to the design that is provided in the E/R diagram in this report.