

INTERNATIONAL UNIVERSITY

VNU-HCM

**Principles of Database Management**

**GROUP PROJECT 7**

**Topic 28: Online voting system database**

**Instructor:** Assoc. Prof. Nguyen Thi Thuy Loan

|  |  |
| --- | --- |
| **Members’ name** | **Student ID** |
| Nguyễn Hoàng Lý | ITITIU20249 |
| Nguyễn Trần Trung Kỳ | ITITIU20240 |
| Văn Bảo Khánh | ITITWE19026 |
| Nguyễn Trần Hoàng Nam | ITTIWE18013 |
| Huỳnh Văn Vĩnh | ITITIU19248 |

**Project Analysis**

This project aims to implement a database for the “online voting system” program. The system should allocate a thread for every new incoming voter.

**Design and Implementation**

1. A secure system that allows users with an authentic name and password to cast a vote
2. Voters are registered by admin, and the voter list is stored in the Online Voting database.
3. The system can match the client name and password to the text file.
4. If the information matches, the voter is redirected to the voting site.

**Table of content**

[**1. Introduction. 4**](#_xjsp1w9mjpf9)

[1.1 Problems. 4](#_jbwmzrex262c)

[1.2 Objectives. 4](#_2hs8mhlabuk6)

[**2. Requirements 4**](#_wuxjkvy4u59g)

[2.1 Tools used 4](#_6j6zb2wmefzk)

[2.2 Librabry 5](#_qa7igeilcs62)

[**3. Expected results. 6**](#_y3bxutecrfdq)

[**4. Implement the plan. 7**](#_hj23ecs69a2w)

[**5. Basic concepts of ERD 7**](#_r0l167icybdl)

[5.1 Notation 7](#_hhrbs0v6ep3k)

[5.2 Main points 8](#_wdlvlbd0d9kx)

[5.3 Symbols in Online Voting 8](#_ljyiyfxbv0df)

[5.4 ERD to Relation Model 9](#_7biqtkclzwjb)

[**6. Software Proof of Concept 11**](#_ol7urmadndsw)

[6.1 Objective 11](#_im9qud61duut)

[6.2 Structured Query Language 11](#_s8lq9qk0rcv8)

[6.3 Back-end 11](#_cuvy3jpymx2d)

[6.4 Front-end 15](#_8d9cozjm63rt)

[6.5 Sample Data 20](#_f5lt38tcbbmq)

[**7. Conclusion 22**](#_bzpc4wfah7ju)

# **1. Introduction**.

## 1.1 Problems.

An online voting system is a software platform that allows groups to securely conduct votes and elections. High-quality online voting systems balance ballot security, accessibility, and the overall requirements of an organization's voting event.

At their core, online voting systems protect the integrity of your vote by preventing voters from being able to vote multiple times. As a digital platform, they eliminate the need to gather in-person, cast votes using paper, or by any other means (e.g. email, insecure survey software).

## 1.2 Objectives.

The goal of our team's online voting system is to create software that can help voters easily vote securely and accurately. Besides, voters can easily view information, positions, and related information about cadidates who will participate in the election.

# **2. Requirements**

## 2.1 Tools used

| **Database Administration Task** | **Preferred Database Tool** | **Other Database Tools** |
| --- | --- | --- |
| Grant database roles | SQL management studio | Oracle Enterprise Manager Console |
| Create a database | SQL management studio | None |
| Insert data | Netbeans | SQL management studio |
| Delete a database | SQL management studio | None |
| Data query | Netbeans | SQL management studio |
| Start a database | SQL management studio | None |
| Shut down a database | SQL management studio | Control Panel |

## 2.2 Librabry

* **Library Requirement:**
  + mssql-jdbc-8.2.2.jre8.jar
  + JDK 19

# **3. Expected results**.

In this project, our team is expected to build an Online voting database with a simple interface to demonstrate the work. The website allows administrators to import numerous candidates, and each voter should register an account full of information to be licensed to vote. Each candidate can be voted many times to reach the highest votes while each voter can only vote once per account.

# 

# **4. Implement the plan**.

1. Member’s task (Overview) :

● Khang makes the front-end (app interface)

● Ly, Ky make backend (java code, sql query)

● Vĩnh, Nam write a report, power point, and other related things that need to be submitted

2. Timetables

|  |  |
| --- | --- |
| Stage (week) | Contents |
| 4 - 5 | Sketch ERD diagram with  description detail (all members) |
| 6-8 | Draw ERD diagram (Vĩnh, Nam) |
| 9 | Switch from ERD diagram to  Entity-Relationship Model (Lý,  Kỳ) |
| 10 | Summary and check again  (Khang) |

# **5. Basic concepts of ERD**

## 5.1 Notation

An Entity Relationship Diagram is a graphical representation that depicts relationships among people, objects, places, concepts or events within a system. ERDs use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes to help us visualize how data is connected in a general way. Because of this, ERDs are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research.

## 5.2 Main points

Our database is designed to be able to manage votes from voters for candidates, so the main point of our database is the tables containing personal information about voters and candidates. in which, voters and candidates are connected to each other through the admin table and the vote table.

## 5.3 Symbols in Online Voting

|  |  |
| --- | --- |
| Symbols | Meanings |
|  | Entity |
|  | Attribute |
|  | Relationship |
|  | Weak Entity |

## 

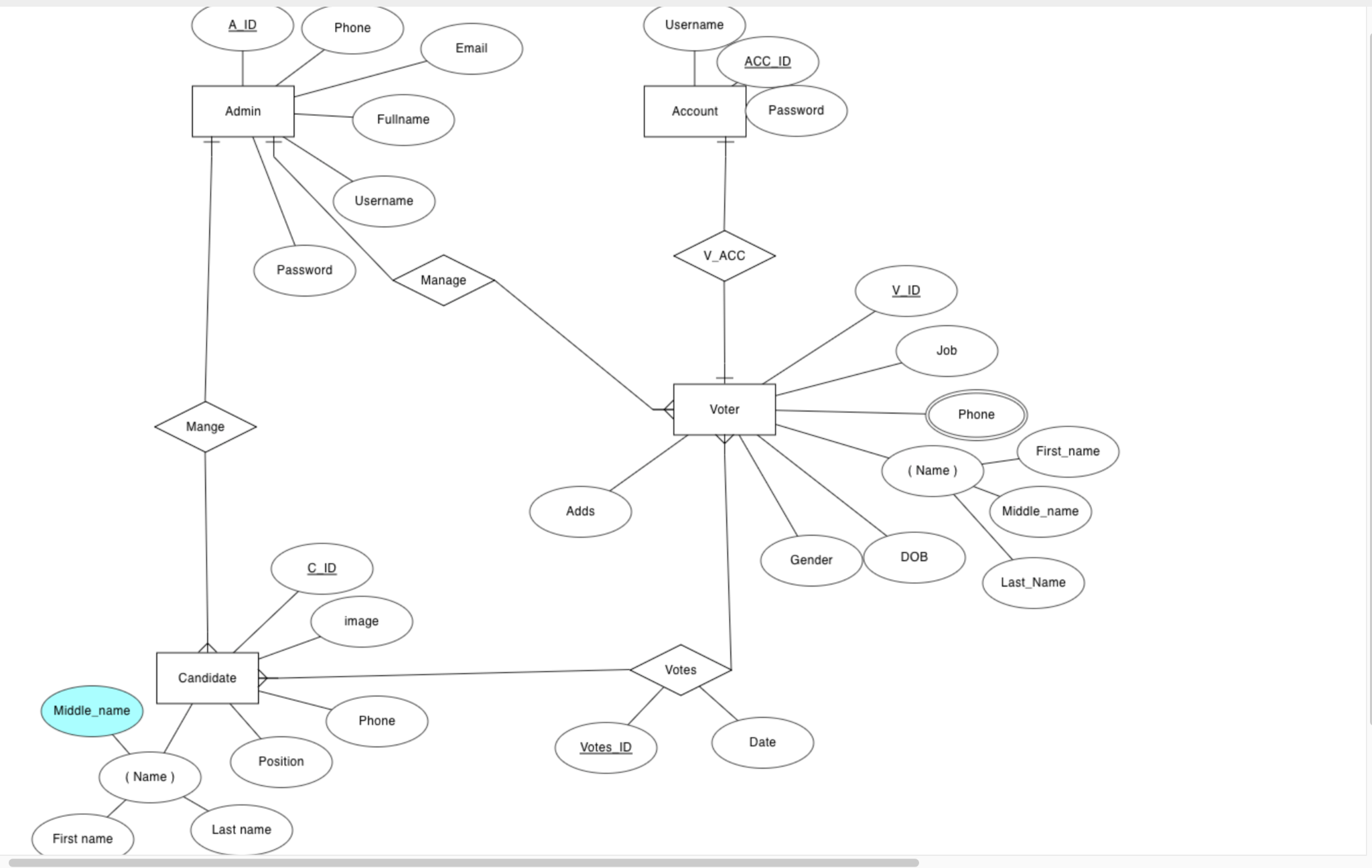


Figure 5.1: ERD Diagram

## 5.4 ERD to Relation Model

Following the first step of the regular entity, we created a corresponding relation that includes simple attributes, choosing one of the key attributes as primary key. Entities Admin, Voter, Candidate, Account.

* Admin = (A\_ID, Phone, Email, Fullname, Username, Password)
* Account = (ACC\_ID, Username, Password, V\_ID)
* Voter = (V\_ID, Job, Phone, Name, DOB, Gender, First name, Last name, Middle name, A\_ID)
* Candidate = (C\_ID, First name, Last name, Middle name, Phone, Position, A\_ID)
* Votes = (Votes\_ID, Date, V\_ID, C\_ID)
* Voter\_Phone = (Phone, V\_ID)

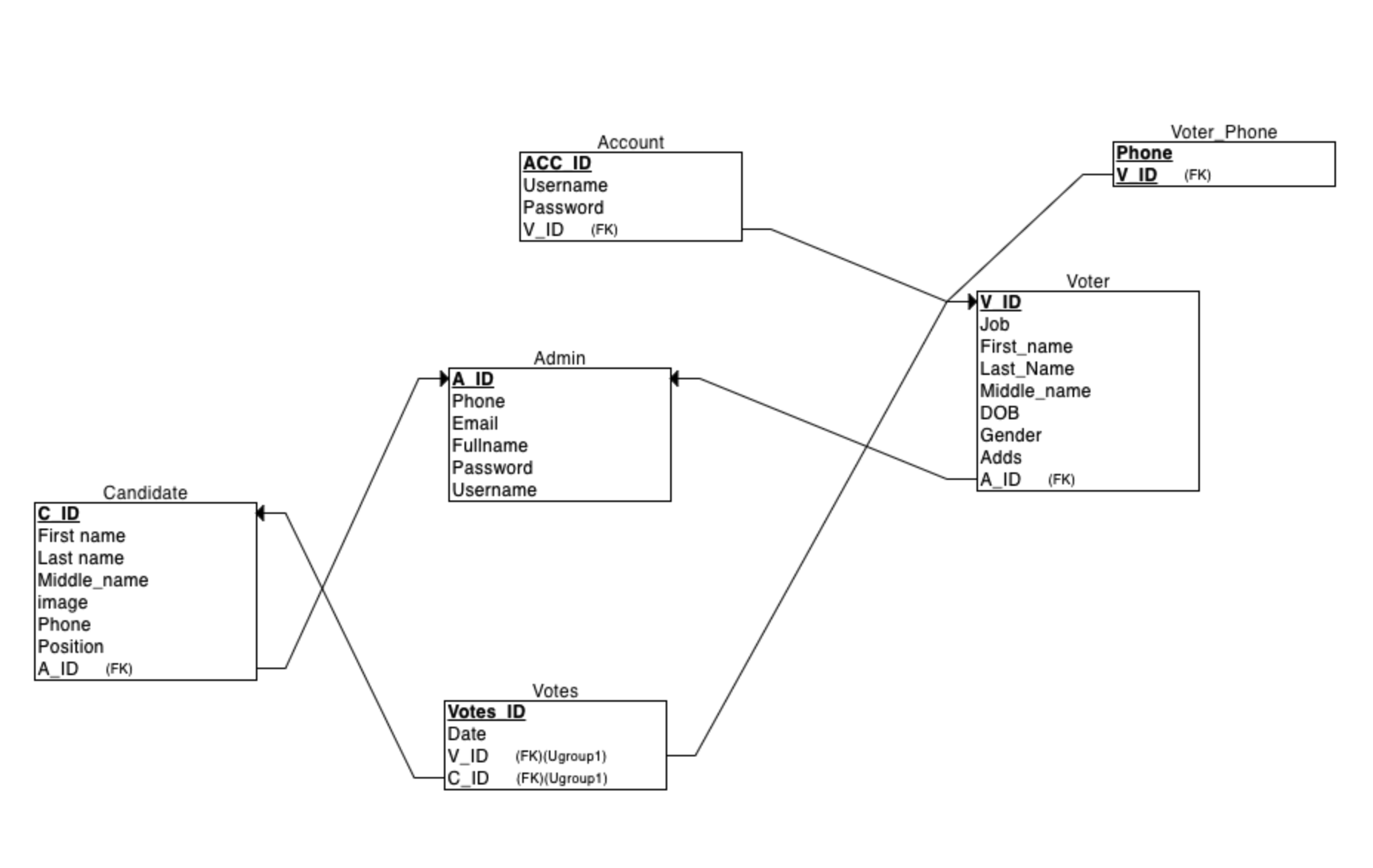


Figure 5.2: Relational Model

# 

# **6. Software Proof of Concept**

## 6.1 Objective

This chapter will describe the basics of the software for using and the notion of Structured Query Language, as well as presenting a basic program with back end and front end. In the back-end, we present how to connect databases from interfaces connected with MySql. About, Front end we use HTML and CSS to build interfaces.

## 6.2 Structured Query Language

Testing the most common use cases to most complex use cases to ensure a solution will deliver the BI and analytics results needed, while fitting the criteria for speed, performance, cost efficiency, and security & governance.

## 6.3 Back-end

Our project is designed in java which is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. Using it connects to the server in software SQL Server.

Firstly, create a database with SQL Management Studio corresponding to relational schema

As a next step, it is essential to develop a user interface function and establish seamless integration with the database system. We divided this project into 4 packets: Admin, Connection, Forms, Voter.

1. Admin: Admin plays the main roles in this system, including adding candidates and voter, view results, and even adding other adminstrators.

**AddminPanel class**: The AddminPanel class has a simple function of displaying a panel with four options: adding a voter, adding a candidate, adding an admin, and viewing the results. However, the standout feature is the "View results" option, which requires selecting the correct position and returning the candidate with the highest number of votes for that position. To better illustrate this feature, we write code that executes a query to retrieve a table containing the candidate with the most votes for their respective position, including their first name, last name, and position attribute.

"SELECT v.Position, c.First\_name, c.Last\_name, MAX(v.Votes\_count) AS MaxVotes FROM (

SELECT Votes.C\_ID, Candidate.First\_name, Candidate.Last\_name, Candidate.Position, COUNT(\*) AS Votes\_count

FROM Votes

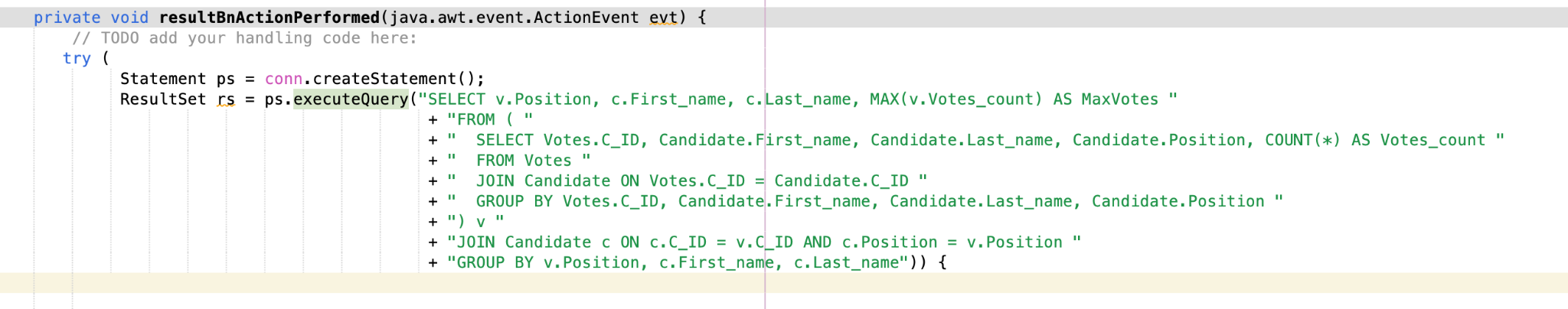
JOIN Candidate ON Votes.C\_ID = Candidate.C\_ID

GROUP BY Votes.C\_ID, Candidate.First\_name, Candidate.Last\_name, Candidate.Position

) v

JOIN Candidate c ON c.C\_ID = v.C\_ID AND c.Position = v.Position

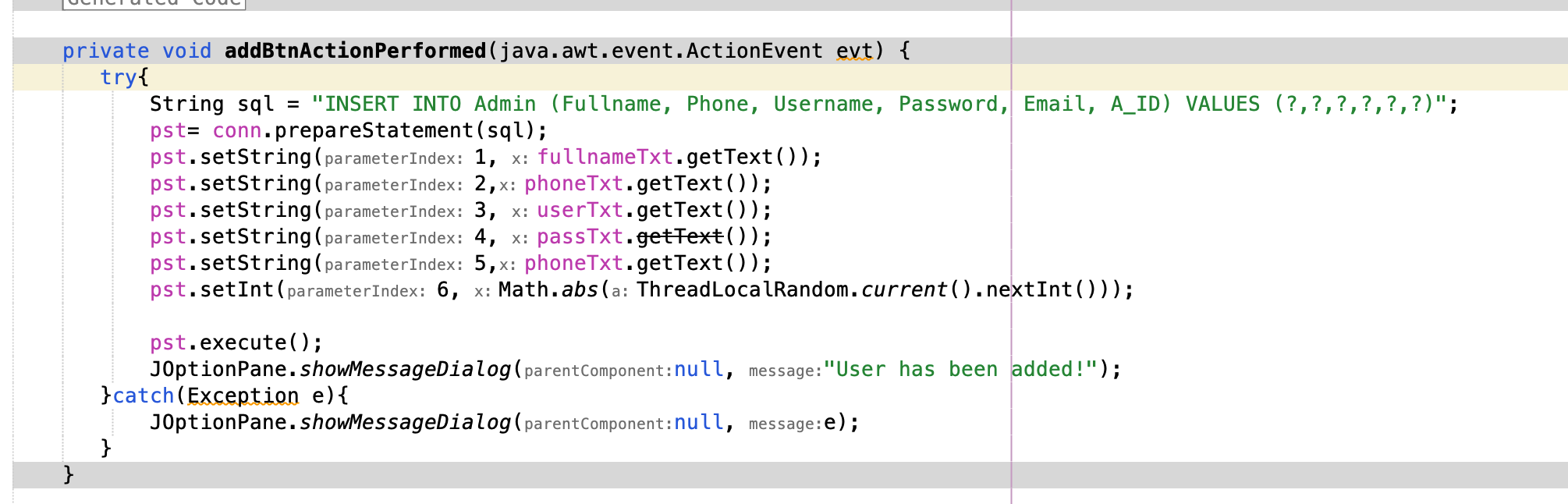
GROUP BY v.Position, c.First\_name, c.Last\_name"



Then, push the data into each block in the ViewReport window.

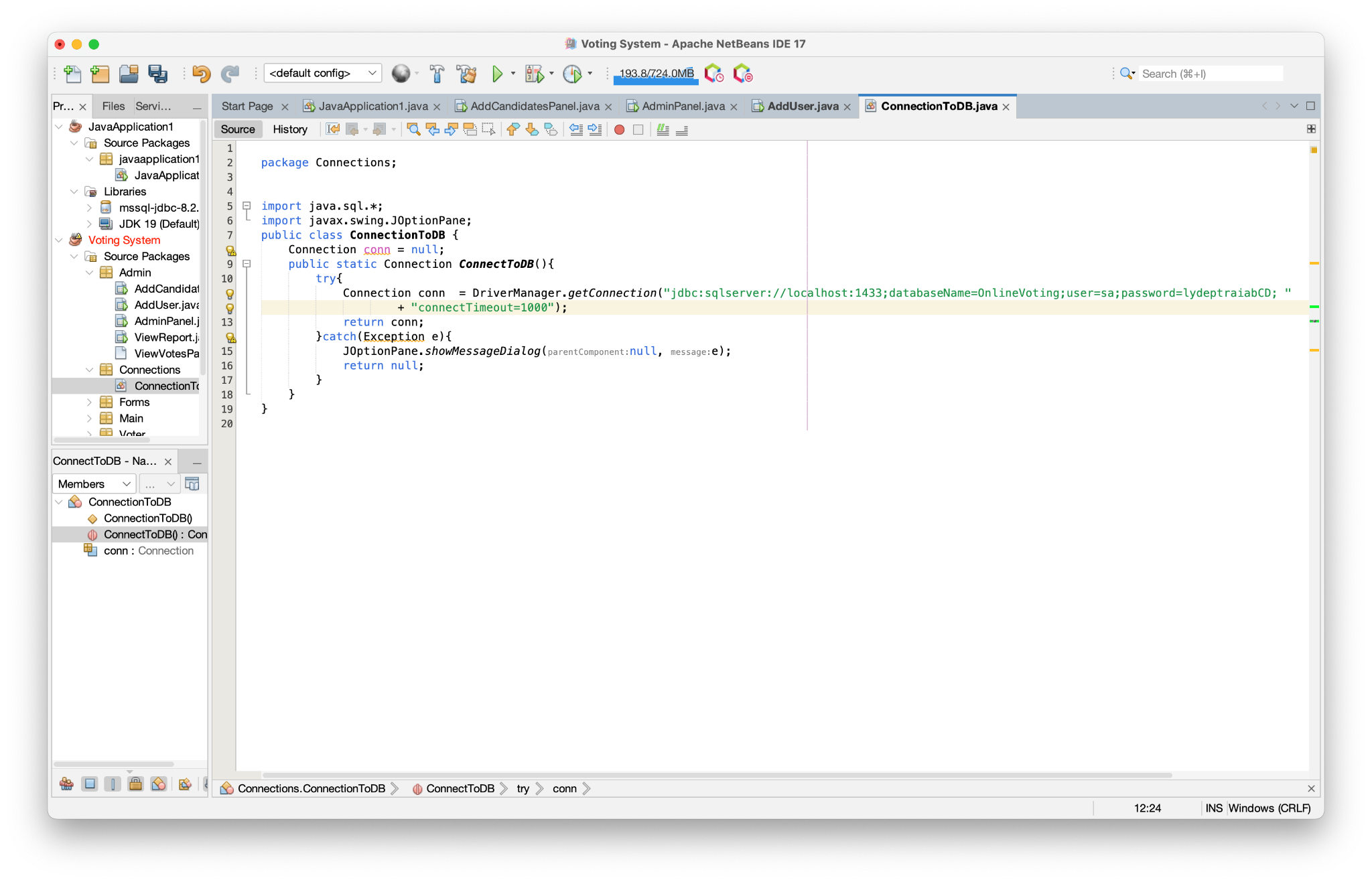
**AddCandidatePanel class**: When the "add" button is pressed, the system will verify that the admin has provided complete candidate information and a valid admin ID using the query statement "SELECT A\_ID FROM Admin WHERE A\_ID =?". If the query does not return a valid ID, the system will not proceed. However, if the ID is valid, the system will insert the candidate using the query statement "INSERT INTO Candidate (C\_ID, First\_name, Last\_name, Phone, Position, Votes\_count, A\_ID, Middle\_name) VALUES (?,?,?,?,?,?,?,?)". It is worth noting that the symbol "?" represents the value received from the input block text.

**AddUser class**: When the "add" button is pressed, the system will execute the query statement “INSERT INTO Admin (Fullname, Phone, Username, Password, Email, A\_ID) VALUES (?,?,?,?,?,?)” to add a new administrator. The admin’s id is randomly generated with Math.abs(ThreadLocalRandom.current().nextInt()). It is worth noting that the symbol "?" represents the value received from the input block text. Note that the symbol “?” is the value received from the in block text

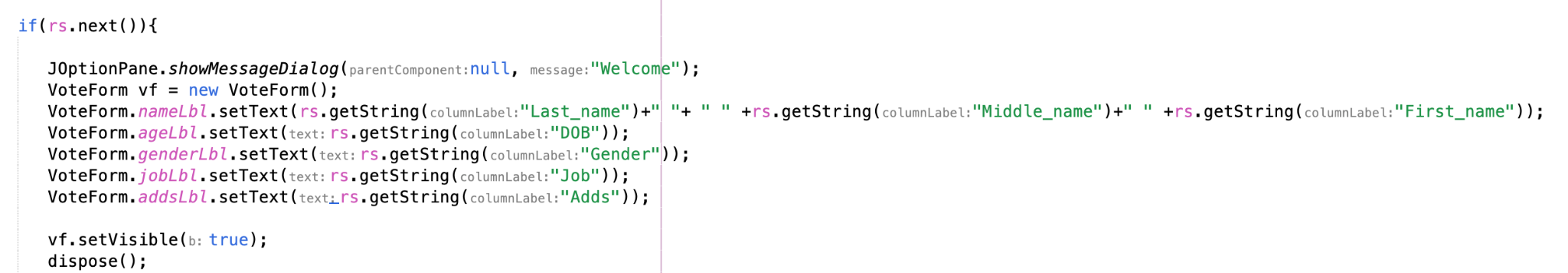


**ViewReport class**: this class simply is a window to display the results of voting, the information is pushed from AddCandidatePanel class.

1. Connection:The ConnectToDb.java class contains a ConnectToDB() method that enables the user to establish a connection with a database and interact with it. This class plays a crucial role in connecting to the database, and it is important to input the correct username and password for your SQL server to ensure a successful connection.



1. Forms

**Login class**: this class plays as the lobby of system because it will display first and require user login to determine who they are an administrator or voter. Firstly the system will check which option user choose “Admin” or “Voter”. If “Voter”, the system will get the Voter’id from Account entity to verify the Voter with query statement *“SELECT V\_ID FROM Account WHERE Username='"+userTxt.getText()+"' AND Password='"+passTxt.getText()+"’”.* Then, jump to the Votes form and push the information of the voter into Votes From . If “Admin”, the system will execute the query statement “*SELECT Username,Password FROM Admin WHERE Username = ? AND Password = ?”*. If there is any result, the admin is vitrified and jumps to the admin panel.

**QueryForm**: This is an independent class to query. Please input the username and password of the SQL server on your computer again.

1. Voter

**AddVoter class**: The system will check that the admin’sID is verified, unless the admin cannot add a voter. Then, the system will execute query statement *“INSERT INTO Voter (V\_ID, Last\_name, First\_name, DOB, Gender, Adds, Job, A\_ID, Middle\_name) VALUES (?,?,?,?,?,?,?,?,?)”*. Besides, creating an account owned by that voter with the query statement *“INSERT INTO Account (ACC\_ID, Username, Password, V\_ID) VALUES (?,?,?,?\_)”*.

**VoterForm class**: This class is very complex because when voters vote for a candidate corresponding to a position, the system will update the vote count in that candidate, and create a votes with date and the ID of the voter and candidate. To be more specific, when the voter clicks the “vote” button, the system will retrieve the votes count of candidate when that candidate is picked with query statement *“SELECT Votes\_count FROM Candidate WHERE First\_name='"+stringName[0]+"' AND Middle\_name='"+stringName[1]+"' AND Last\_name='"+stringName[2]+"' AND position='Secretary’”*. Then plus that vote count by 1 and save it in a variable for example *“secretaryVote”*. Then, create a votes with the query statement *“INSERT INTO Votes (Votes\_ID, Date, V\_ID, C\_ID) VALUES (?,?,?,?)”*, and update the vote count with the query statement *“UPDATE Candidate SET Votes\_count='"+ secretaryVote +"' WHERE First\_name='"+stringName[0]+"' AND Middle\_name='"+stringName[1]+"' AND Last\_name='"+stringName[2]+"' AND position='Secretary'”*

1. Connections: The ConnectToDb.java class contains a ConnectToDB() method that enables the user to establish a connection with a database and interact with it. This class plays a crucial role in connecting to the database, and it is important to input the correct username and password for your SQL server to ensure a successful connection.

## 6.4 Front-end

Because of learning a course of using JFrame Form in Netbean IDE, we decided to work with it to design our front-end. JFrame Form is a tool which allows developers to design UI without any codes. This IDE’s GUI Builder makes it possible to build professional-looking GUIs without an intimate understanding of layout managers. You can lay out your forms by simply placing components where you want them.

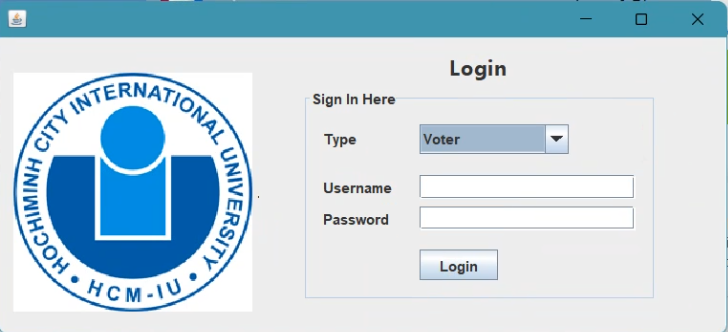


Figure 6.1: Login Page

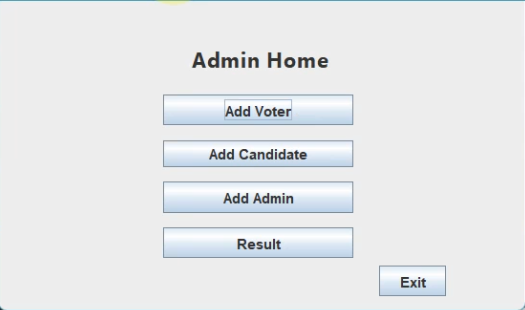
****

Figure 6.2: Admin Page

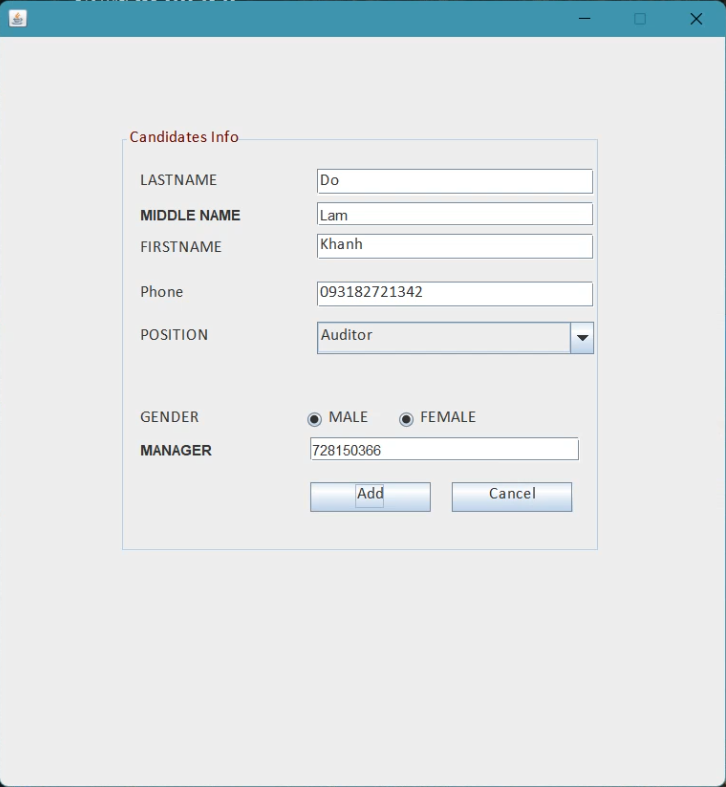


Figure 6.3: Add Candidates Form

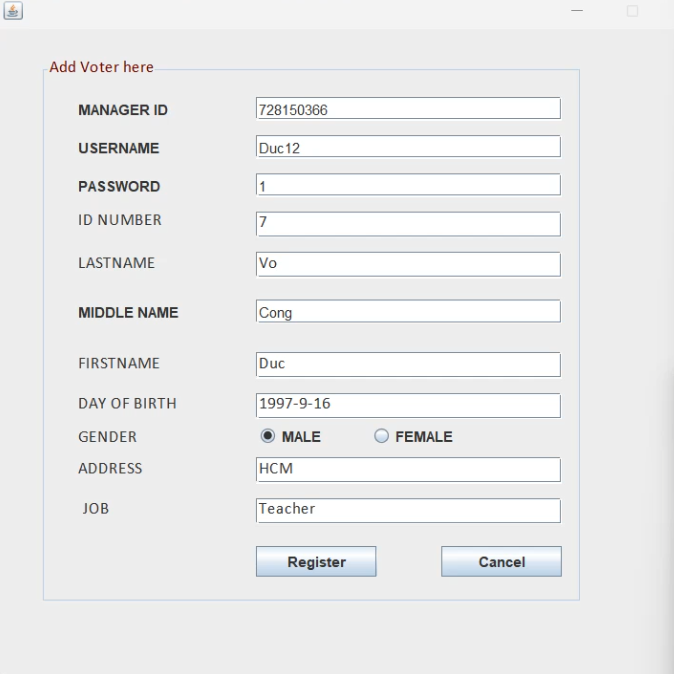


Figure 6.4: Add Voter Form

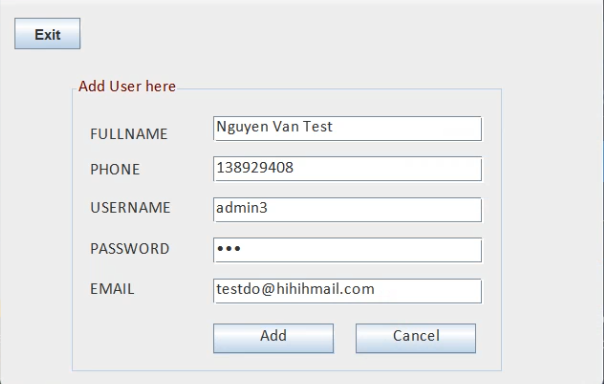


Figure 6.5: Add New Admin

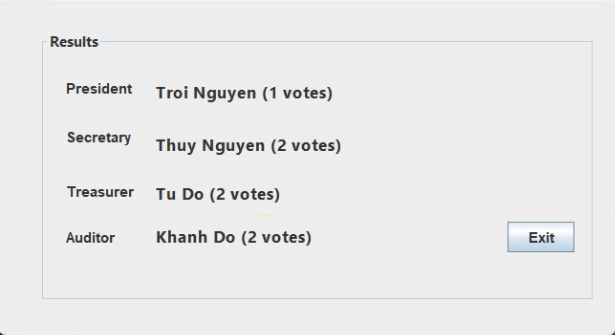
****

Figure 6.6: Voting Result

## 6.5 Sample Data

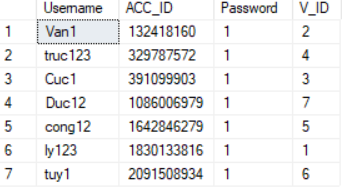


Figure 6.1: Account data sample

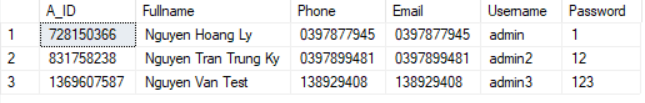


Figure 6.2: Admin data sample



Figure 6.3: Candidate data sample

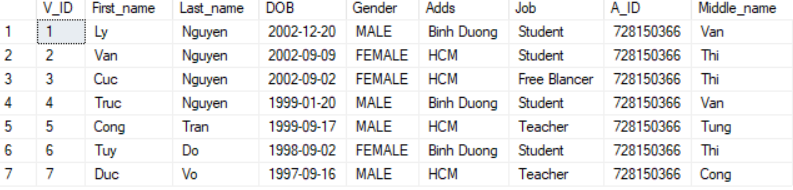


Figure 6.4: Voter data sample

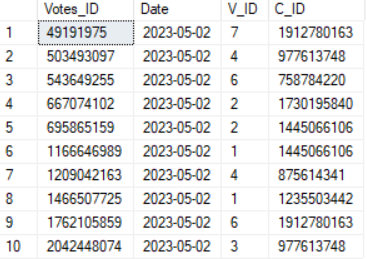


Figure 6.5: Votes data sample

# 

# **7. Conclusion**

Overall, the Online Voting system is needed for every purpose. With secured information of voters and verified candidates, the user needs to register and sign in the system, then pick a category and choose the candidates within the category.

**8. Query Question**

**1. Find the candidates who have the most votes**

SELECT TOP 10 Candidate.First\_name, Candidate.Last\_name, COUNT(Votes.V\_ID) AS VoteCount

FROM Candidate

JOIN Votes ON Candidate.C\_ID = Votes.C\_ID

GROUP BY Candidate.C\_ID, Candidate.First\_name, Candidate.Last\_name

ORDER BY VoteCount DESC;

A screenshot of a computer

Description automatically generated with medium confidence

Explain: This SQL query selects the first name, last name, and vote count for the top 10 candidates with the most votes. The JOIN clause combines the Candidate and Votes tables on the C\_ID foreign key, and the GROUP BY clause groups the results by candidate ID, first name, and last name. The COUNT function counts the number of votes for each candidate, and the ORDER BY clause sorts the results in descending order by vote count. Finally, the TOP keyword limits the results to the top 10 candidates.

**2. Find the number of voter who vote more than 3 times**

SELECT COUNT(DISTINCT Votes.V\_ID) AS NumVoters

FROM Votes

GROUP BY Votes.V\_ID

HAVING COUNT(\*) > 3;

Explain: This query selects the distinct Vid values from the Votes table and groups them by Votes.Vid. It then uses the COUNT function to count the number of times each voter appears in the table. The HAVING clause filters the results to only include groups where the count is greater than 3. Finally, the COUNT function is used again to count the number of distinct voters who meet the criteria.

**3. Find the voters who live in Main St**

SELECT Voter.\*

FROM Voter

WHERE Voter.Adds LIKE '%Main St%';

A screenshot of a computer

Description automatically generated with low confidence

Explain: This query selects all columns from the Voter table and filters the results to only include rows where the Adds column contains the string "Main St". The LIKE operator is used with the % wildcard to match any string that contains "Main St", regardless of whether it appears at the beginning, middle, or end of the address.

4. **Find the candidate who have the smallest votes**

5. Find all the candidate live in [address] trong candidate có cho dia chỉ đâu mà kêu tìm

6. Find the voter who votes [candidate names] follow by first\_name & last\_name

7. Find the numbers of voter/candidates live in every city

8. Find the candidate that have the highest votes follow by name & address

**9. Find the first name of the voters who are male**

SELECT First\_name

FROM Voter

WHERE Gender = 'male';

A screenshot of a computer

Description automatically generated with medium confidence

This query selects the First\_name column from the Voter table for all rows where the Gender column is equal to 'male'. The WHERE clause filters the results to include only rows where the Gender column has the value 'male'.

**10. Find the voter who voted recently**

SELECT TOP 1 Voter.\*

FROM Voter

JOIN Votes ON Voter.V\_ID = Votes.V\_ID

ORDER BY Votes.date DESC;

A screenshot of a computer

Description automatically generated with medium confidence

Explain: This query selects all columns from the Voter table for the voter who voted most recently. The JOIN clause joins the Voter and Votes tables on the V\_ID foreign key, and the ORDER BY clause sorts the results in descending order by the date column in the Votes table. The TOP 1 clause limits the results to the first row, which corresponds to the voter who voted most recently.

Online Voting Source Code: <https://drive.google.com/drive/folders/1LDLDizHy9qtnUsxD4TRLR8vxgaIWUgwj?usp=sharing>

Slides:

<https://www.canva.com/design/DAFh3AlX0Ys/94_U_U1F-XZRpzNA56JZKg/edit?utm_content=DAFh3AlX0Ys&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton>