**JavaFX Project**

**🎓E-Class System**

# Project Members:

## SULEYMAN Jumaniyazov      EG6X6K

## ALBERTO Balazs Sandor       H64GOG

# Supervisor

## Dr. Subecz Zoltan

****

**John Von Neumann University**

**Academic Year: 2024-2025**

**Introduction with Thanks**

We are immensely grateful to the divine presence that has granted me the fortitude, vitality, and determination to accomplish this endeavor.

Our heartfelt gratitude extends to my beloved family members whose unwavering support and various forms of assistance have propelled me to this juncture of our educational journey.

We extend our sincere appreciation to Dr. Subecz Zoltan, our esteemed supervisor, whose invaluable guidance, insightful feedback, and unwavering encouragement have been instrumental in bringing this project to fruition.

We express profound gratitude to the esteemed members of the evaluation committee for graciously agreeing to assess our work.

Lastly, We extend heartfelt thanks to all the educators who have imparted their knowledge and wisdom, as well as to all individuals who have supported and assisted us throughout my academic and professional pursuits.

**INSTRUCTION**

1. On the first page, present the company on a spectacular website **(2 points)**
2. Have Registration, Login option **(3 points)**

- The "Login" menu item is visible if the user is not logged in.

- The "Logout" menu item is visible when the user is logged in.

- Display the logged-in user on the system header, if logged in

1. Distinguish at least 3 user roles: **(3 points)**

Admin, User, Visitor

The appearance of the menu items and the availability of the pages vary depending on

which user is using the site. (e.g. admin page)

1. Have a page where you display data from the chosen database **(2 points)**

Use the data of 3 tables from the database.

1. One page should have a contact form, which you can use to send a message **(2 points)**

and can be sent to the site owner. Check the form correct filling with

server-side validation. Save the sent form data to the database.

1. Make it possible to view previous point messages on a table from the database. **(2 points)**

in reverse chronological order. Print the sending time and the sender name

for each message. In the case of a non-logged-in user: "Guest".

1. Implement a RESTful API in the application. **(3 points)**

Test API functionality with both cURL and Postman. Take screenshots

of the tests to the documentation

1. Use the GitHub (github.com) version control system. **(2 points)**

**(Mandatory element! The source will be checked based on it)**

1. Use the project work method on GitHub: **(2 points)**

Display which part of the project was prepared by which group member.

Don't just upload the finished application in one step, but also the partial states

in at least 5 steps.

**Project Overview**

The EClass System is a Spring Boot-based web application designed to manage educational data. It features functionalities including user registration and authentication, database management, and CRUD operations for various entities.

### Team Collaboration

**Team Members**

* 🧑‍💻 Suleyman Jumaniyazov
* 🧑‍💻 Alberto Magarin

**Work Distribution**

* **[Student 1** Suleyman Jumaniyazov**]** **🧑‍💻**
* Implemented database management system
* Developed CRUD operations for Students and Subjects
* Created data import and export utilities
* Implemented statistics and reporting features
* **[Student 2** Alberto Magarin**]** **🧑‍💻**
* Developed MNB Exchange Rates service
* Implemented Parallel Processing demo
* Created data visualization components
* Handled UI design and user interaction

**Technical Architecture**

**Technologies Used**

* Java 17
* JavaFX
* SQLite
* Apache POI
* SLF4J Logging

**Key Features**

1. **Database Management 📂**
   * ➕🔍✏️❌CRUD operations for Students, Subjects, and Marks
   * Data import from text files
   * Advanced filtering and searching
2. **MNB Exchange Rates 💱**
   * 🌐Real-time currency rate download
   * 📈 Dynamic Line chart visualization
   * 🗓️Flexible date and currency selection
   * 💹 Multiple currency support
3. **Parallel Processing 🔀**
   * Demonstrates concurrent thread execution ⏱️
   * Two independent counters with different update intervals 🕰️
   * Thread-safe implementation 🧵

**Database Structure 💾**

**Tables**

1. **students**
   * id (INTEGER PRIMARY KEY) 🆔
   * sname (TEXT)
   * class (TEXT)
   * boy (INTEGER)
2. **subjects**
   * id (INTEGER PRIMARY KEY) 🆔
   * sname (TEXT)
   * category (TEXT)
3. **marks**
   * id (INTEGER PRIMARY KEY) 🆔
   * studentid (INTEGER)
   * mdate (TEXT)
   * mark (INTEGER)
   * type (TEXT)
   * subjectid (INTEGER)

**Project Setup and Deployment**

**Prerequisites**

* Java Development Kit 17
* Maven
* SQLite

**Building the Project**

bash

mvn clean package

**Running the Application**

bash

java -jar target/eClassSystemJavaFX-1.0-SNAPSHOT.jar

**GitHub Repository**

* Repository Link: [Insert GitHub Repository URL]

**🧩 Development Workflow**

**1. 🎬 Project Initialization**

* Create project structure
* Set up Maven dependencies
* Configure JavaFX environment

**2. 🗃️ Database Setup**

* Design SQLite schema
* Implement data import mechanisms
* Create CRUD service classes

**3. 🖥️ UI Development**

* Design FXML layouts
* Implement JavaFX controllers
* Create interactive components

**4. 🔌 Service Integration**

* Develop MNB Exchange Rates service
* Implement parallel processing demo
* Add error handling and logging

**5. 🧪 Testing and Refinement**

* Unit testing
* Integration testing
* User experience optimization

**🚧 Challenges and Solutions**

**1. 🧵 Thread Synchronization**

**Challenge:** Ensuring thread-safe updates in parallel processing **Solution:**

* Used **AtomicInteger** for counter management
* Implemented **Platform.runLater()** for UI updates

**2. 📡 Real-time Data Fetching**

**Challenge:** Retrieving dynamic exchange rates **Solution:**

* Asynchronous thread-based data retrieval
* Implemented robust error handling
* Fallback mechanism for unavailable data

**🔮 Future Roadmap**

* 🔐 User authentication system
* 📈 Advanced analytics dashboard
* 🌐 Cloud synchronization
* 📱 Mobile companion app

**💡 Key Learnings**

* Modular application design
* Concurrent programming techniques
* JavaFX UI development
* Database management best practices

**📂 Root Project Structure**

eClassSystemJavaFX/

│

├── src/

│ ├── main/

│ │ ├── java/ # Java source code

│ │ │ └── com/

│ │ │ └── example/

│ │ │ └── eclasssystem/

│ │ │ ├── Main.java # Application entry point

│ │ │ ├── services/ # Core business logic

│ │ │ ├── controller/ # UI controllers

│ │ │ ├── model/ # Data models

│ │ │ └── util/ # Utility classes

│ │ │

│ │ ├── resources/ # Non-code resources

│ │ │ ├── fxml/ # JavaFX layout files

│ │ │ ├── database/ # SQLite database

│ │ │ ├── txt/ # Initial data files

│ │ │ └── css/ # Styling resources

│ │ │

│ │ └── module-info.java # Java module configuration

│ │

│ └── test/ # Unit and integration tests

│

├── pom.xml # Maven project configuration

└── DOWNLOAD.zip # Distributable package

**🔍 Key Architectural Components**

**1. 🚪 Entry Point: Main.java**

A computer screen shot of a program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

**A computer screen shot of a program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A screen shot of a computer

Description automatically generated**

1. **🎨 Controller Design Pattern**

Architectural Highlights:

Reactive UI updates

Separation of data loading and presentation

Dynamic filtering and searching

**A computer screen shot of text

Description automatically generated**

1. **🔀 Parallel Processing Demonstration**

Concurrency Techniques:

ExecutorService for thread management

AtomicBoolean for thread synchronization

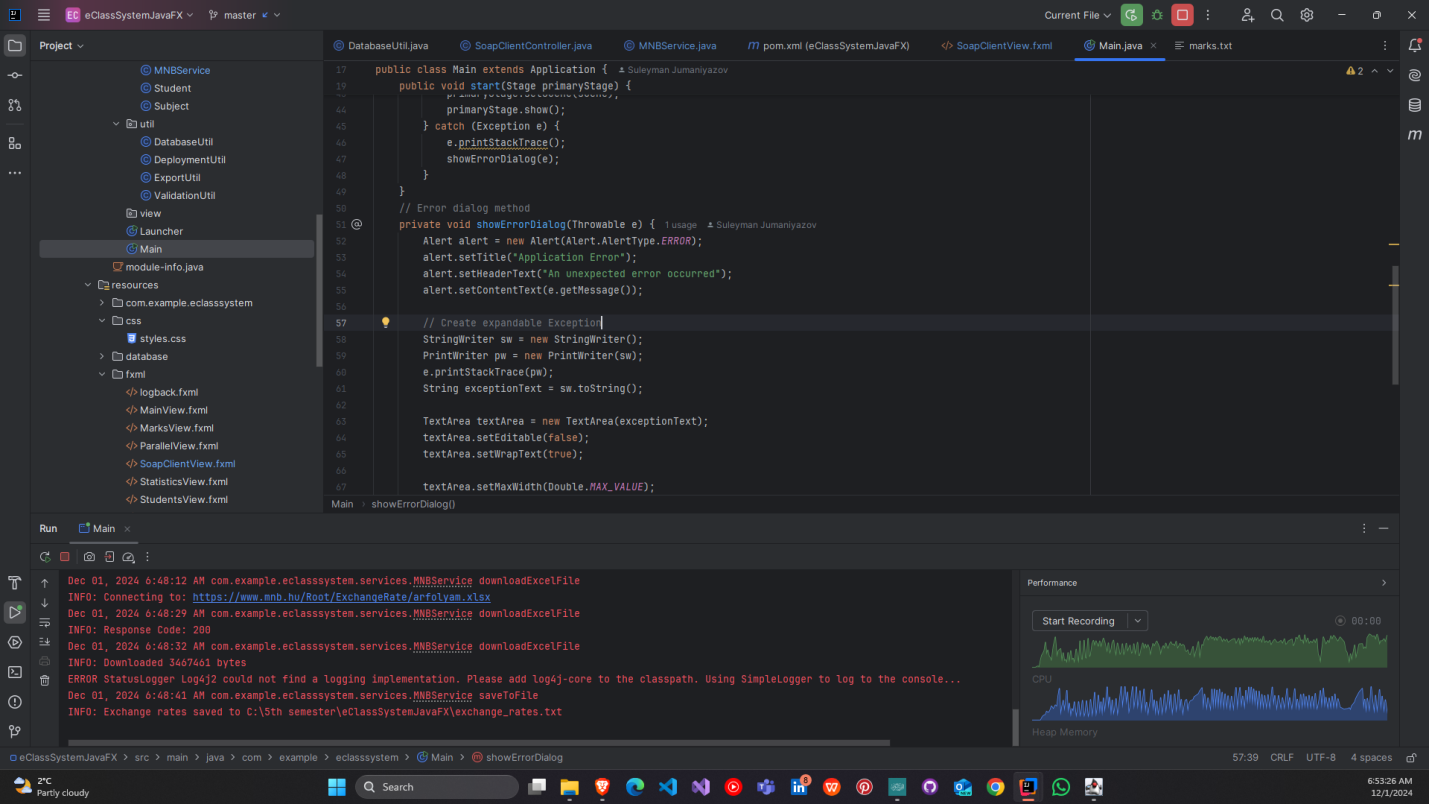
Platform.runLater() for safe UI updates

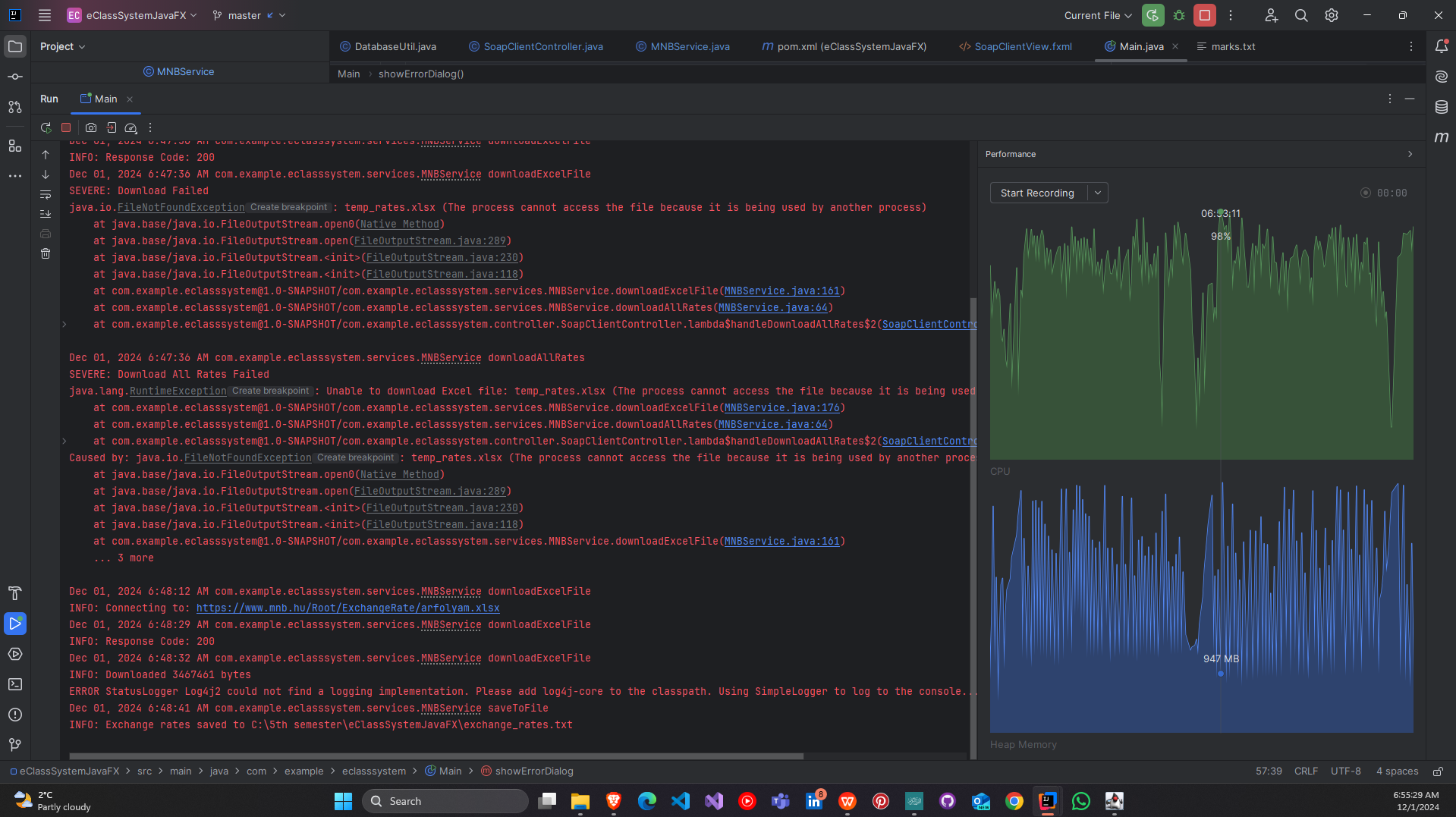
**A computer screen shot of a program code

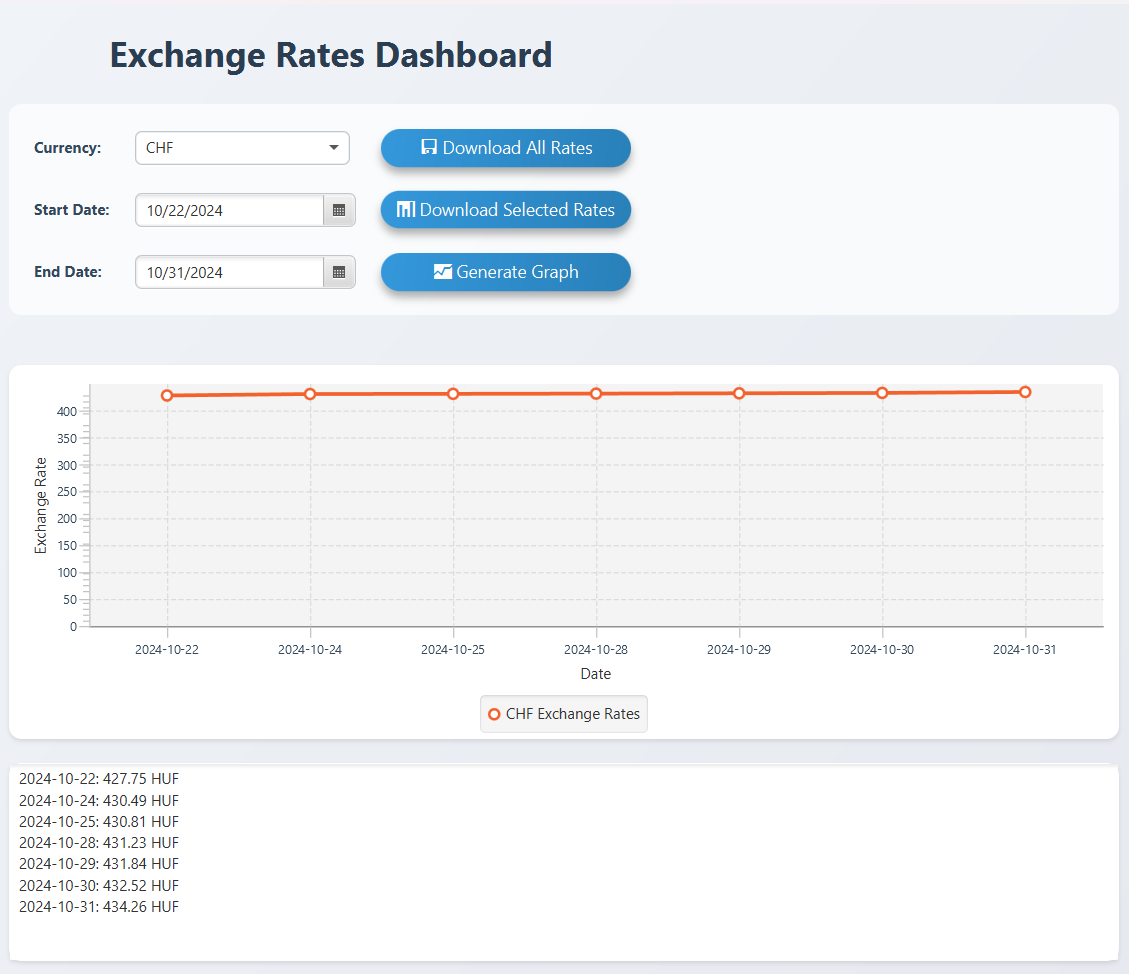
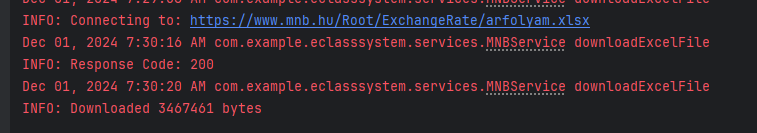
Description automatically generated**

**Here I tried to download all data, and it took me very long time as It was very comprehensive way to fetch data From MNB banks, so please in testing select maximum 2 month period but better just 10 days please, becaouse it takes long time as in the back end there is a lot of fetching analizing functions whci in my PC takes 2-5 minutes.**

**Here in below is way to testing inside the IDE**







**Eventually it is working,**

**So, Now Detailed explanation, if you woud like to see detailes, they are below**

**Excel File Reading Mechanism**

**Purpose**

**The Excel file reading mechanism is designed to efficiently extract and process data from Microsoft Excel spreadsheets using Apache POI library.**

****

****

**These 2 files are crucial algorithm for solving this task**

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

A computer screen shot of a program

Description automatically generated

A screen shot of a computer program

Description automatically generated

**As u saw these are perfectly downloading required or all datas.**

**And Controller is here** A screen shot of a computer program

Description automatically generated

A computer screen shot of a program code

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a program

Description automatically generated

**For JAR FILE**

A screen shot of a computer program

Description automatically generated

And creating JAR FILE

A screenshot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

**AND its front end is like this**

A screenshot of a cell phone

Description automatically generated

A screenshot of a computer

Description automatically generated

Filter is working and as well other functions too A screenshot of a computer

Description automatically generated

A graph with red lines

Description automatically generated

**Marks are soo a lot that why it need to wait a bit,**

**But Subject is not so a lot so it works so fast, I advise to test her** A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

**And its updated directly and we can see**

**A white table with black text

Description automatically generated**

**Optional features downloading as css or pdf and excel**

A screenshot of a computer

Description automatically generated

A close-up of a computer screen

Description automatically generated

A list of subject items

Description automatically generated

**This is PDF file.**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**As u see they are working good**

**And Good UI exchange rates view**

A screenshot of a computer

Description automatically generated

**Please select maximum 1 month as it wont fit grapht and will be bad design, and will take long time**

**For example like this if its small window** A screenshot of a computer

Description automatically generated

Soon we will update it to good UI

**But If Full Screen** A screenshot of a computer

Description automatically generated **like this**

**THIS Jar file is located in DOWNLOAD-MAIN.ZIP folder in girhub**

**Thanks for time and attention.**