CMPT 305 Nobel Prize Explorer Project

Group 3: Jack Staples, Brett Anderson, Mark Tigchelaar, Jordan Hewko

Contributions of each team member

- Jack Staples - Worked on creating the Database class, Queries class, returning Queries, writing getter methods for the laureate class, started the UndoRedo manager, helped interface the backend with the front end, helped with bug finding and fixing, implemented threading for the picture finder, and wrote this report.
- Mark Tigchelaar – Wrote the laureate class, as well as its subclasses; PrizeData, BirthData, DeathData, MiscData. Wrote the container class Laureates that collects the raw CSV from the API and converts it to laureate class. Helped complete work on the UndoRedo Manager.
- Jordan Hewko – Wrote and designed the class responsible for generating the pictures. Helped convert the code to implement the UI. Attempted to create a map that would display all search results, but this didn't make it into the final implementation. Orientated the User Interface to be more aesthetically pleasing. Also implemented the reset fields button.
- Brett Anderson – Created the graphical user interface, that interacts with the backend to display the laureate information. As well implemented the Undo/Redo. Implemented error checking through the UI to catch incorrect input. Created the LaureateController, and LaureateModel classes, as well as helped Jordan with the Layout.
- Everyone – provided input on the design for both the data structures of the front end, and the Graphical elements of the back end.

Good Aspects of this Project

- Offline functionality – by generating a database that is local to the user, you do not have to be connected to the internet to search through the laureates once the application has started. The only functionality it loses is picture grabbing.
- Threading – The picture grabbing section of the application was causing the whole app to hang while it generated, connected to, and handled the response to the picture API. To solve this, we create a thread for the pictures so that the information can be displayed while the picture is searched for.
- Speed of Queries– By generating an offline database and using maps to sift through much of the data, queries are returned within milliseconds of a request being made. This is much faster than connecting to the Nobel API for each request.
- Code Reuse – On the backend many functions are written to be used by different functions to accomplish similar goals. A good example of this is the addField method that is used when adding Laureates to the database. It inserts a given laureate attribute (eg, first name, last name, gender, category) along with their ID into a given Map for later querying.
- Search by Names – Another advantage of generating a database is that we can search laureates by first and last names. The Nobel API does not allow for this functionality. Our engine uses regex strings to filter out users, so if a user wants they can search by pattern.

- Resilience – The input from users is handled in such a way that if they give strange or erroneous input, the application should handle it rather than crash.
- Scalability – The application is designed in such a way that it should continue to function when new users are added to the Nobel API. The new winners will show up in the system.

Drawbacks

- Time on Startup – They system takes some time to launch, because it is first communicating with the Nobel API for the raw CSV, generating laureate classes from that CSV, and then sorting those class objects into a database. We found that it is only a few seconds, and it really fluctuates based on internet speed.
- Must be Connected on Startup – If a user is not connected to the internet when they start the application, it will fail to grab the CSV, and a database will not be created. This causes the application to crash.
- No Handling for New Laureates – If a new laureate is added to the API while the application is running, it will not be updated until the application is restarted.
- Double Winner Overwrite – If someone has won twice the laureate object overwrites in the database when they are found a second time. They will still show up when they are searched for, for either award, but it will only show the information for the second win.
- Picture Grabber – Some names have different protocol for the building the URL won't work, because they would have to be hard coded. Also, most comittees won't work for the same reason.