Replicate Antikythera Group

Professor Marisha Rawlins

Applied Programming Concepts

August 2021

Replicate Antikythera

Motivation/ Context

Antikythera was originally a handheld device used by the Ancient Greeks to track planetary orbit and predict celestial events. It resembled a clock sitting on a mantel. It had knobs and handles on the sides. More like the clock, it had a circular center with hands that rotated. Rather than minutes, the hands displayed celestial time. Similarly, we created an executable that used the technique of time analysis to predict celestial events and track planetary orbit. The file opens and provides the user with a menu. This menu gives the user options to display planets as they orbit or predict a planets orbit period. This planet display is a GUI, and it provides the user with a visual display of where the planets will be at a given time and how fast they travel. The other options on the menu will provide the user with information based on the information they enter the search.

Functionality

The project opens and presents the user with a menu on a UI. Asking them what they would like to do.

If they pick ‘Open Model’, a GUI will open and give them the option to watch planet orbit or predict planets orbit. For the predict you can select month, year, or both and it show you where the planets will be at the time you provided If you hover over the planets while predicting it shows the name of that planet, orbit period, orbit speed, distance from sun, mass, and planet type. The orbit button will show you how fast the planets orbit in relation to each other. Whether you decide to orbit or predict, you must select the reset button before changing to either or. If you do not, you will receive an error message.

If the user picks any of the other options, the UI will continue asking them for relevant information needed for whichever option they chose. The program will take whatever information they provide and sift through the database with it, to find the information you are requesting.

This original menu will continue to present itself, until you are finished with the options. Then you can select exit and the program will end.

Proof of Working

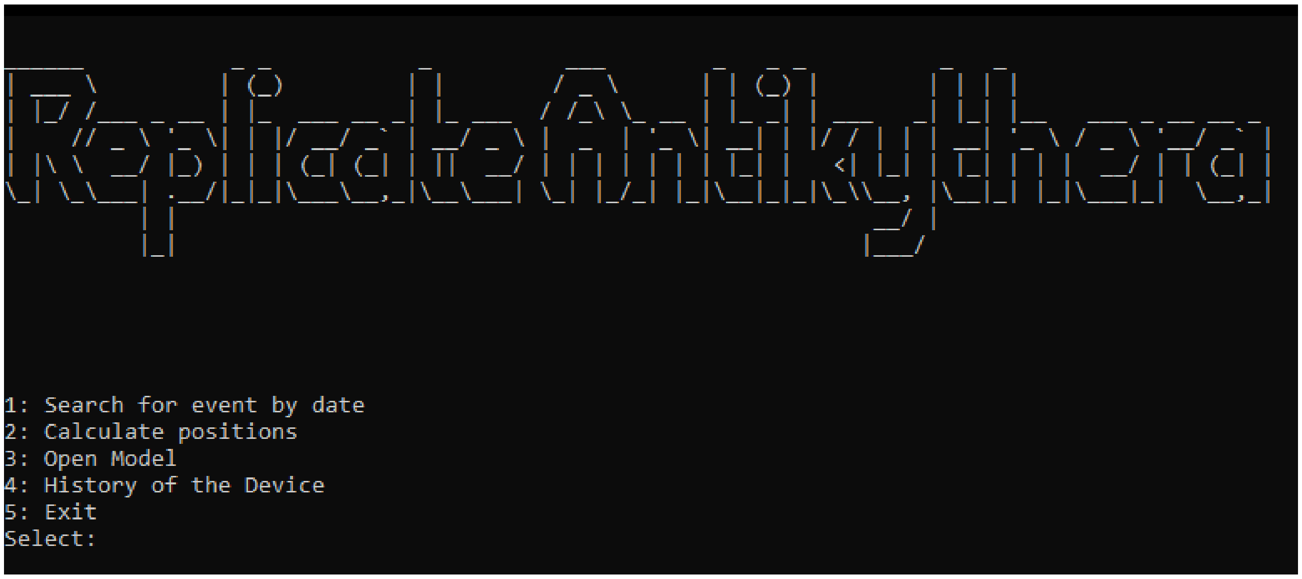
The plan for testing every component for validation is to go through the entire program and test each component as they are used. Opening the main executable to start, the first function to test is the search events by a date. For this function there are two options: to search for an event using the current date, or to search by a specific date on another day.

The next component to test is the prediction feature from the main executable. To test this, the user will enter in several rotations around the sun and the formulas that were used for this feature will return values for the pointers used.

The next component to test is the third option in the code, which is to open the model (GUI). The GUI will undergo specific testing to each of its functions (orbit, predict, display information).

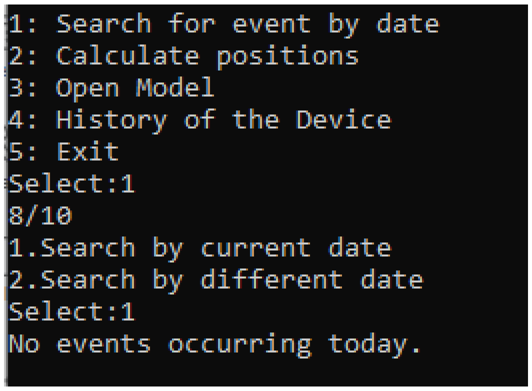
The last function to test is the history of the device option. This will open a link in a default browser to a credited article that talks about the history of the mechanism.

The following captures support the results of running the program to test the individual components as well as the overall system.



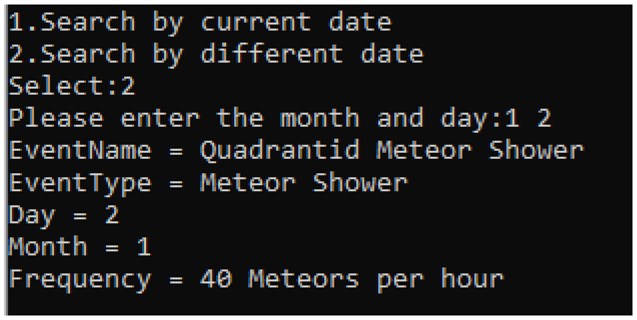
```
______          _ _           _         ___        _   _ _          _   _
| ___ \        | (_)         | |       / _ \      | | (_) |        | | | |
| |_/ /___ _ __ | |_  ___ __ _| |_ ___ / /_\ \_ __ | |_ _| | ___   _| |_| |__   ___ _ __ __ _
|    // _ \ '_ \| | |/ __/ _` | __/ _ \|  _  | '_ \| __| | |/ / | | | __| '_ \ / _ \ '__/ _` |
| |\ \  __/ |_) | | | (_| (_| | ||  __/| | | | | | | |_| |   <| |_| | |_| | | |  __/ | | (_| |
\_| \_\___| .__/|_|_|\___\__,_|\__\___\_| |_/_| |_|\__|_|_|\_\\__, |\__|_| |_|\___|_|  \__,_|
          | |                                                  __/ |
          |_|                                                 |___/



1: Search for event by date
2: Calculate positions
3: Open Model
4: History of the Device
5: Exit
Select:
```

Figure 1 – When the program is opened, the main menu options as well as the title of the program are displayed.



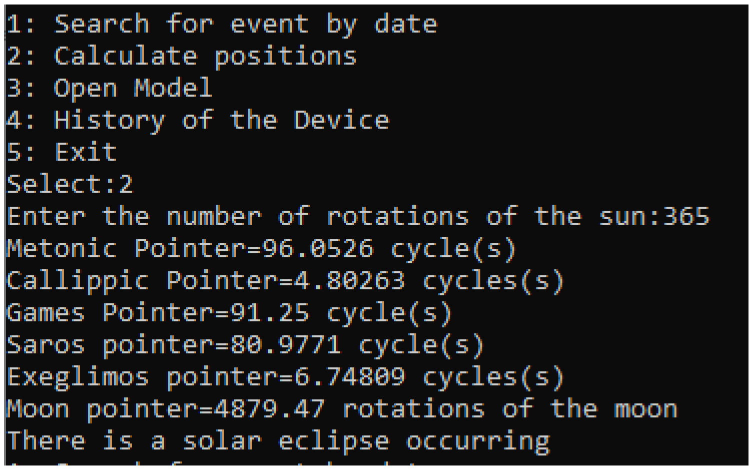
```
1: Search for event by date
2: Calculate positions
3: Open Model
4: History of the Device
5: Exit
Select:1
8/10
1.Search by current date
2.Search by different date
Select:1
No events occurring today.
```

Figure 2 – The first option 'search event by date' is called and user chooses to search by current date by entering a '1' to search for the date 8/10.



```
1.Search by current date
2.Search by different date
Select:2
Please enter the month and day:1 2
EventName = Quadrantid Meteor Shower
EventType = Meteor Shower
Day = 2
Month = 1
Frequency = 40 Meteors per hour
```

Figure 3 – The option to 'search event by date' is chosen and the second option for 'different date' is chosen to search for an event on 1/2, the database was able to pull the event shown above for that date.



```
1: Search for event by date
2: Calculate positions
3: Open Model
4: History of the Device
5: Exit
Select:2
Enter the number of rotations of the sun:365
Metonic Pointer=96.0526 cycle(s)
Callippic Pointer=4.80263 cycles(s)
Games Pointer=91.25 cycle(s)
Saros pointer=80.9771 cycle(s)
Exeglimos pointer=6.74809 cycles(s)
Moon pointer=4879.47 rotations of the moon
There is a solar eclipse occurring
```

Figure 4 – The 'calculate positions' option is chosen, the user is asked to enter a number of rotations around the sun, then the calculations are done to return the values for the pointers shown in the image above.

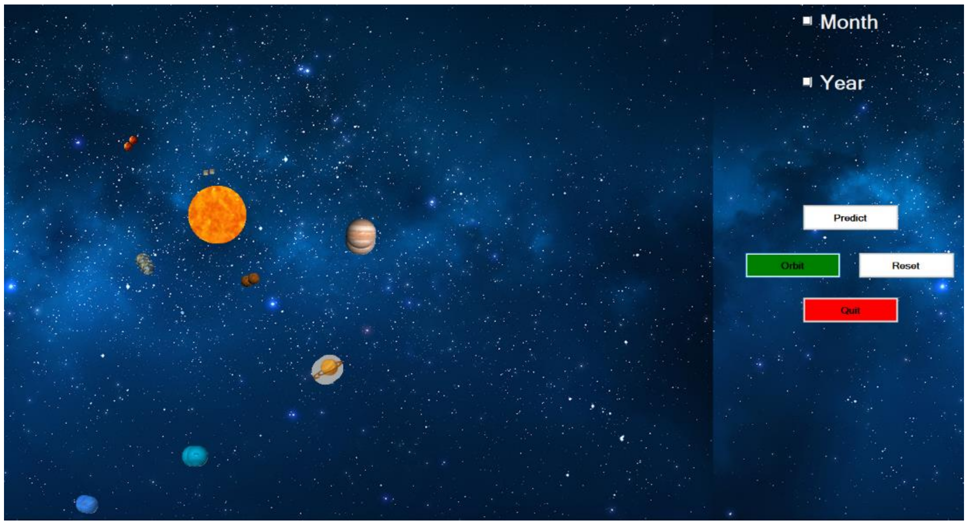
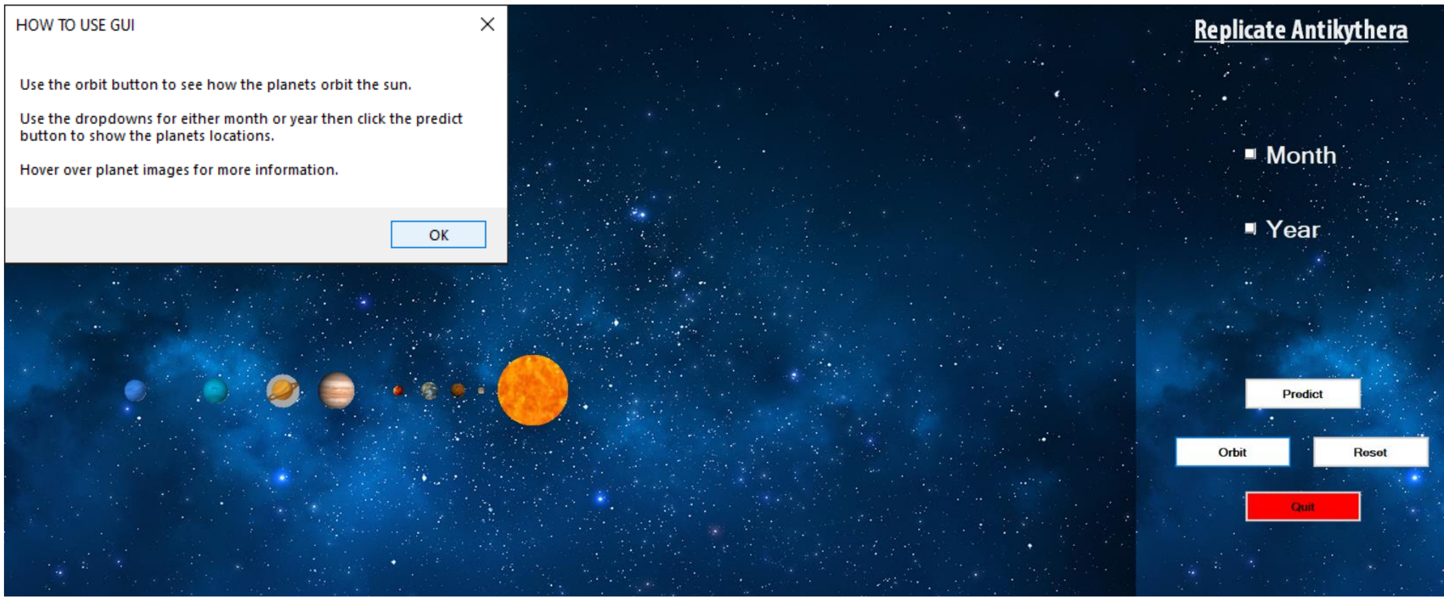


Figure 5 – When the 3rd option is chosen to open the model, the GUI executable is opened, and the window above will pop up on the screen. This portion is to validate that the executable can be opened from the main module.



HOW TO USE GUI

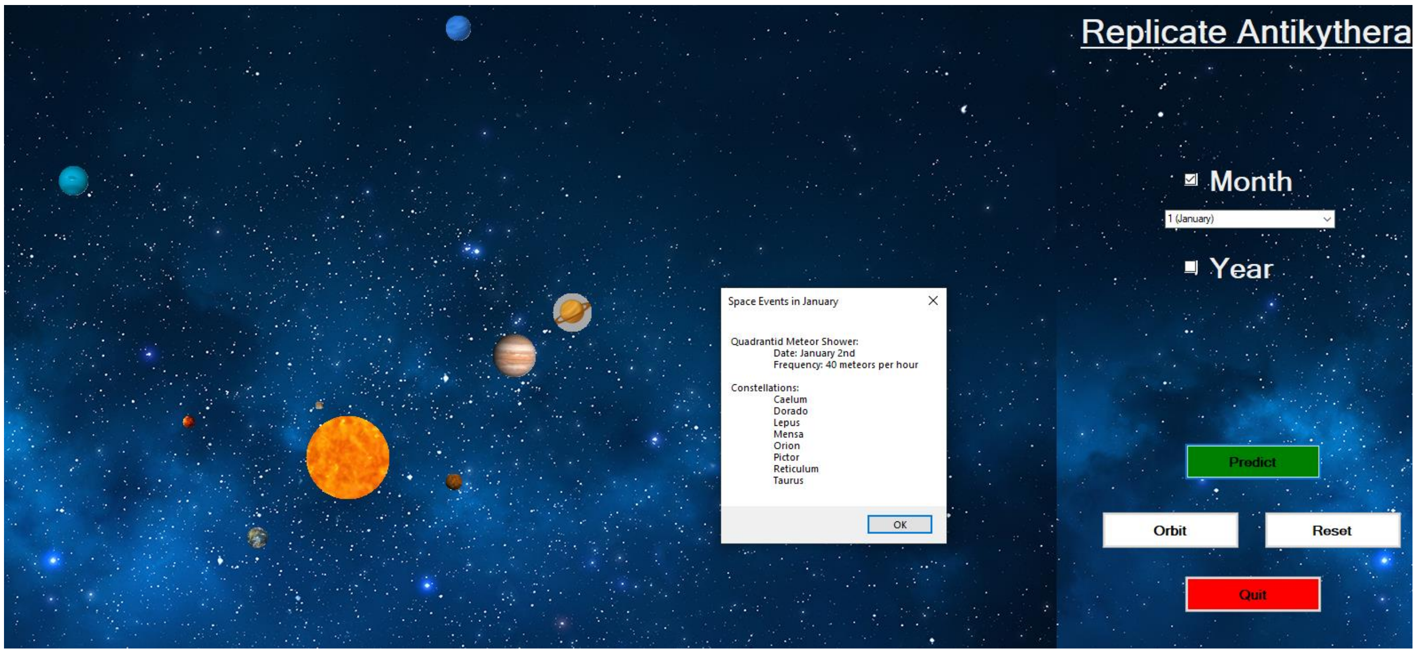
Use the orbit button to see how the planets orbit the sun.

Use the dropdowns for either month or year then click the predict button to show the planets locations.

Hover over planet images for more information.

OK

Figure 6 – The pop up box on the top left of the image above is displayed when the GUI executable is opened, this is proven to work because it appears with the correct information when it is opened.



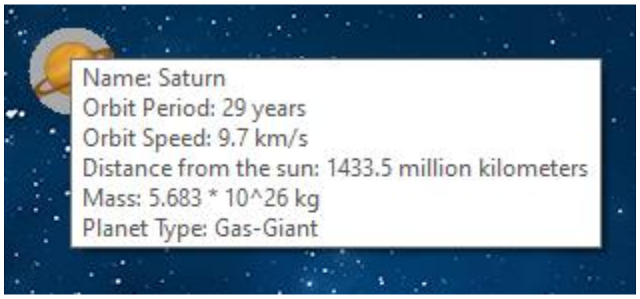
Space Events in January

Quadrantid Meteor Shower:
    Date: January 2nd
    Frequency: 40 meteors per hour

Constellations:
    Caelum
    Dorado
    Lepus
    Mensa
    Orion
    Pictor
    Reticulum
    Taurus

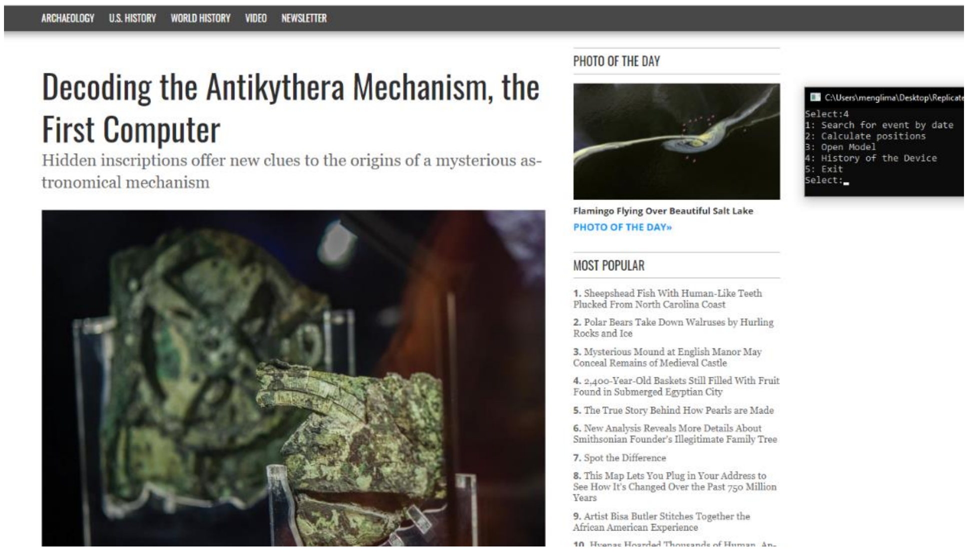
OK

Figure 7 – For the prediction option, there are two checkboxes for month and year. If the user checks off both check boxes then they will be prompted with an error message that they are not allowed do both at the same time. Likewise, if the user tries to orbit and predict at the same time then there will be an error message saying that they are not allowed to do so. The prediction method is used to see where the planets will align at that certain month or year. Based off the comments from the other teams, we added an option so that if the user chooses a month such as January, then they will be prompted with a message box that says all the space events for that specific month. So for January, there is a Quadrantid meteor shower on the date of January 2nd with a frequency of 40 meteors per hour. There are also constellations of Caelum, Dorado, Lepus, Mensa, Orion, Pictor, Reticulum, and Taurus.



Name: Saturn
Orbit Period: 29 years
Orbit Speed: 9.7 km/s
Distance from the sun: 1433.5 million kilometers
Mass: 5.683 * 10^26 kg
Planet Type: Gas-Giant

Figure 8 – The mouse hover function to display planet information is depicted above. When the user hovers over a planet at any time, a pop-up box will display some relative information that is stored for these planets in the database.


ARCHAEOLOGY U.S. HISTORY WORLD HISTORY VIDEO NEWSLETTER

Decoding the Antikythera Mechanism, the First Computer

Hidden inscriptions offer new clues to the origins of a mysterious astronomical mechanism

PHOTO OF THE DAY

Flamingo Flying Over Beautiful Salt Lake

PHOTO OF THE DAY»

MOST POPULAR

1. Sheepshead Fish With Human-Like Teeth Plucked From North Carolina Coast
2. Polar Bears Take Down Walruses by Hurling Rocks and Ice
3. Mysterious Mound at English Manor May Conceal Remains of Medieval Castle
4. 2,400-Year-Old Baskets Still Filled With Fruit Found in Submerged Egyptian City
5. The True Story Behind How Pearls are Made
6. New Analysis Reveals More Details About Smithsonian Founder's Illegitimate Family Tree
7. Spot the Difference
8. This Map Lets You Plug in Your Address to See How It's Changed Over the Past 750 Million Years
9. Artist Bisa Butler Stitches Together the African American Experience
10. Hyenas Hoarded Thousands of Human, An-

```
C:\Users\menglima\Desktop\Replicate
Select:4
1: Search for event by date
2: Calculate positions
3: Open Model
4: History of the Device
5: Exit
Select:_
```

Figure 9 – The above image is of the article that is opened when the 4th menu option, 'history of the device', is chosen. This menu option will open a link to the article shown above that discusses the history of the Antikythera mechanism.

Modeling

Waterfall Model

1. *Requirement definition*: (Total Estimated time of completion – 14 weeks)
    a. Calculate the positioning of stars/planets/etc. (main program with calculation methods)
        i. Find formulas from Antikythera research
    b. Database with the different celestial bodies and their related info
        i. Comets/meteor showers, planets, stars, misc.
    c. User will interact with program in some way to calculate positioning/date/etc. of a desired celestial body
        i. GUI will have some way of giving options for the user to choose what they want to see about a certain celestial event
2. *System and software design*: (estimated time of completion – 6-8 weeks)
    a. User Interface
        i. GUI (Visual studio)
            1. Icon and background image related to space
            2. Use of drop downs, check boxes, buttons
            3. Visuals: planets orbit around the sun with accurate speeds and distances from sun, also the predict feature puts the planet icons in the location in orbit they would be for the parameter

b. Database (estimated time of completion – 3-4 weeks)
    1. Multiple tables for the different celestial bodies (listed attributes)
        a. Constellations: name, month
        b. Planets: name, distance, mass, speed, orbit, period, type
        c. Celestial events: event name, event type, day, month, frequency

c. Main code (methods/variables/etc.)
    i. Menu style UI
        1. Search events by date (current date, or input other)
        2. Calculate planet positions
        3. Open the GUI executable
        4. Link to article with device history

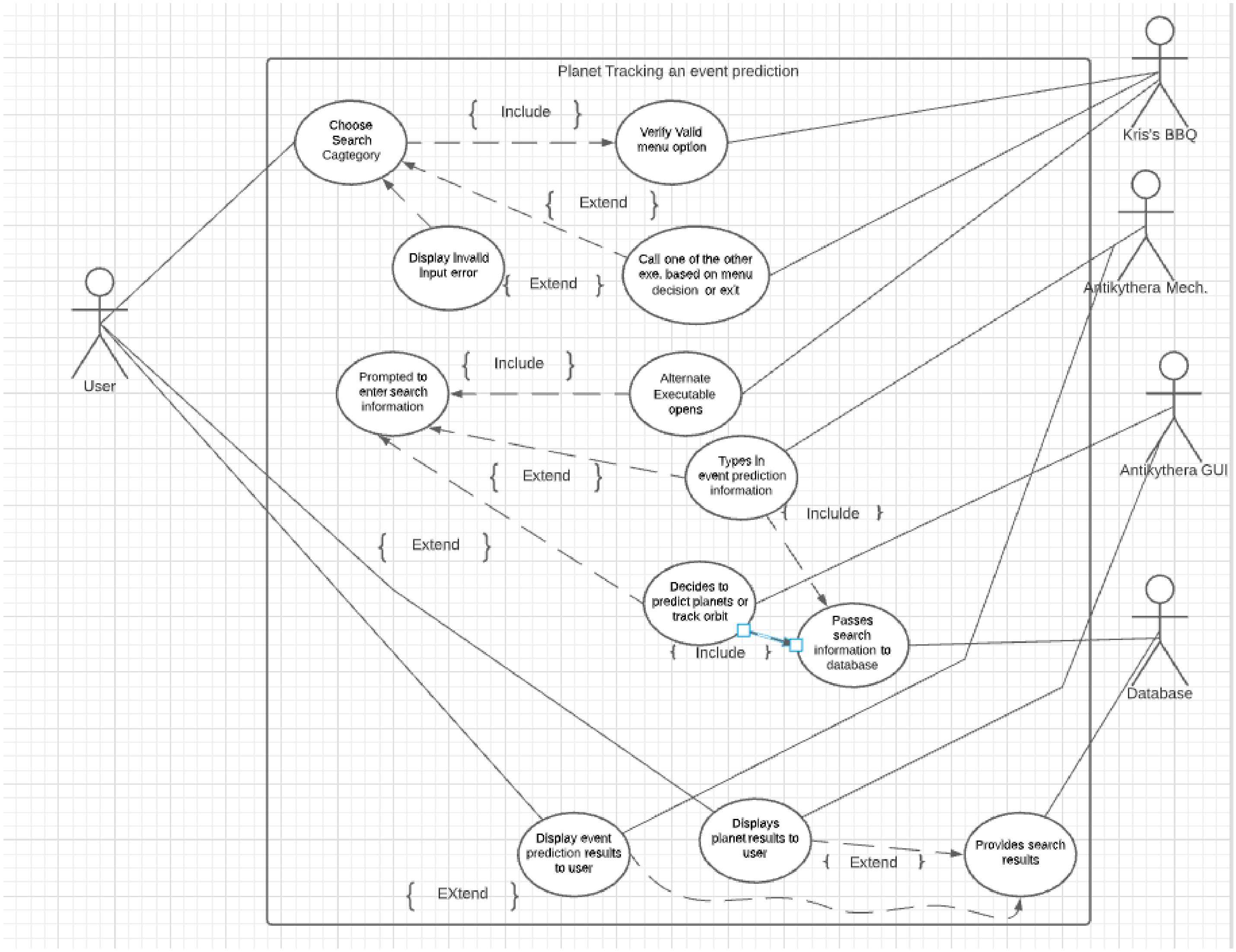
3. *Implementation and unit testing*: (Estimated time of completion for unit testing – 2-3 weeks)
    a. The languages that are used are C++ and C#
    b. The following functions will be tested to assure proper functionality
        i. GUI
            1. Ensure the planets orbital paths in visuals are accurate
            2. Test the predict functions to make sure the planet icons are in the correct position
            3. Make sure all buttons and drop downs work and meet requirements
        ii. Main code
            1. Test position calculations to ensure the formulas used are accurate
            2. Test the open executable command
            3. Test the command to open link to article
            4. Test the events on a date to ensure the data is being pulled from database
        iii. Database
            1. Make sure all data can be pulled correctly from the needed tables
            2. Also ensure that no data is lost from database when commits are made
4. *Integration and system testing*: (Estimated time of completion for maintenance – 1-2 weeks

The integration and system testing phase includes taking the different source codes and getting a functioning executable. Once the entire GUI and the main menu style UI are tested throughout for all functions that a user can perform, the main product is tested. This will include running the program as an ordinary user would and checking for any more bugs that were missed after the first testing phase. One key function of the program that needs to be tested is opening the GUI executable from the main code. The use of running the program on different machines will be crucial to ensure that when the system is being used by other groups, it will work as expected.

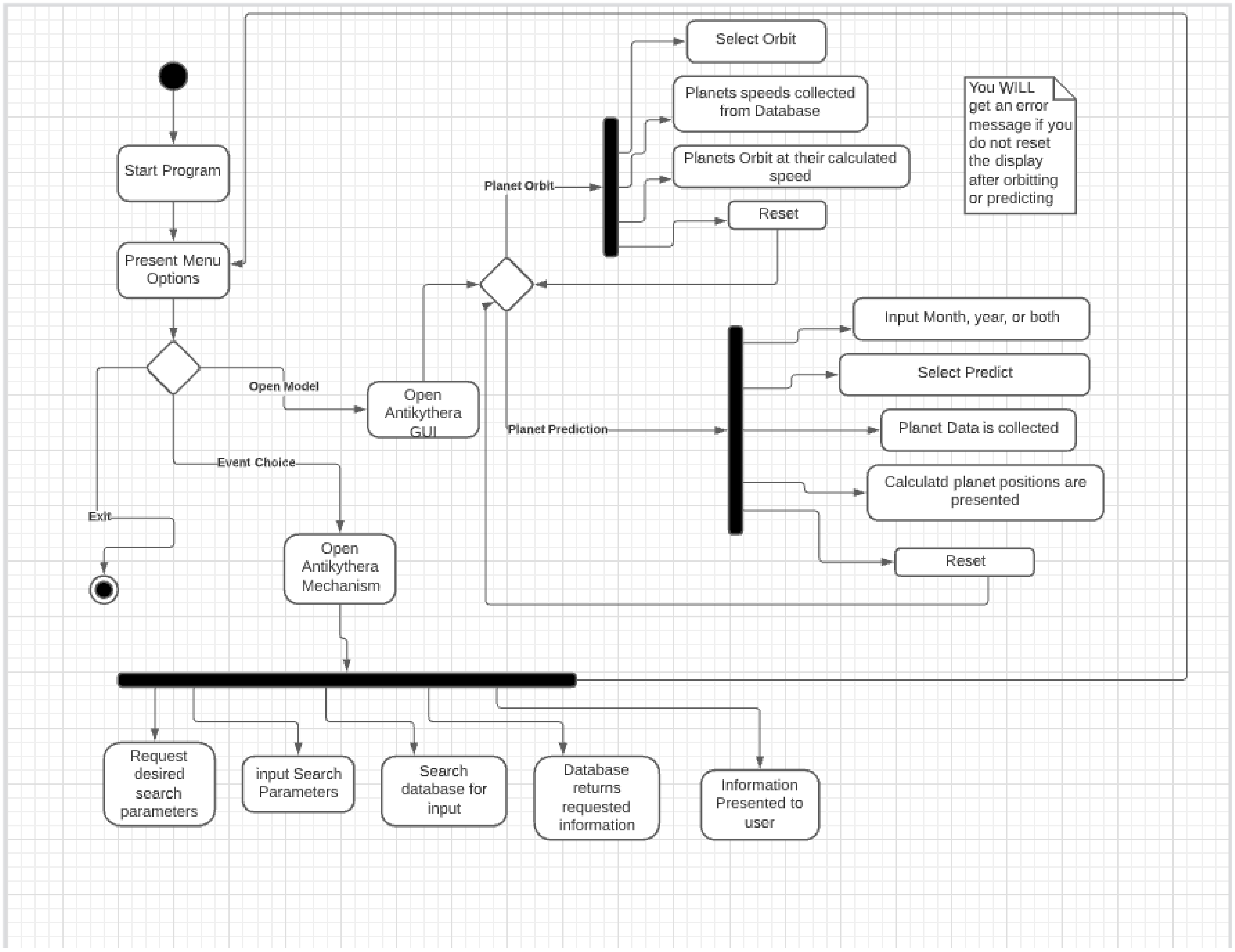
5. *Operation and maintenance*: (Estimated time of completion for maintenance – 1-2 weeks)

After the testing stages, the group will receive feedback from the other groups that will be taken into consideration for further bug fixes and other features to add. This time will be taken to put finishing touches on the program and add features that were not implemented in the beta version.

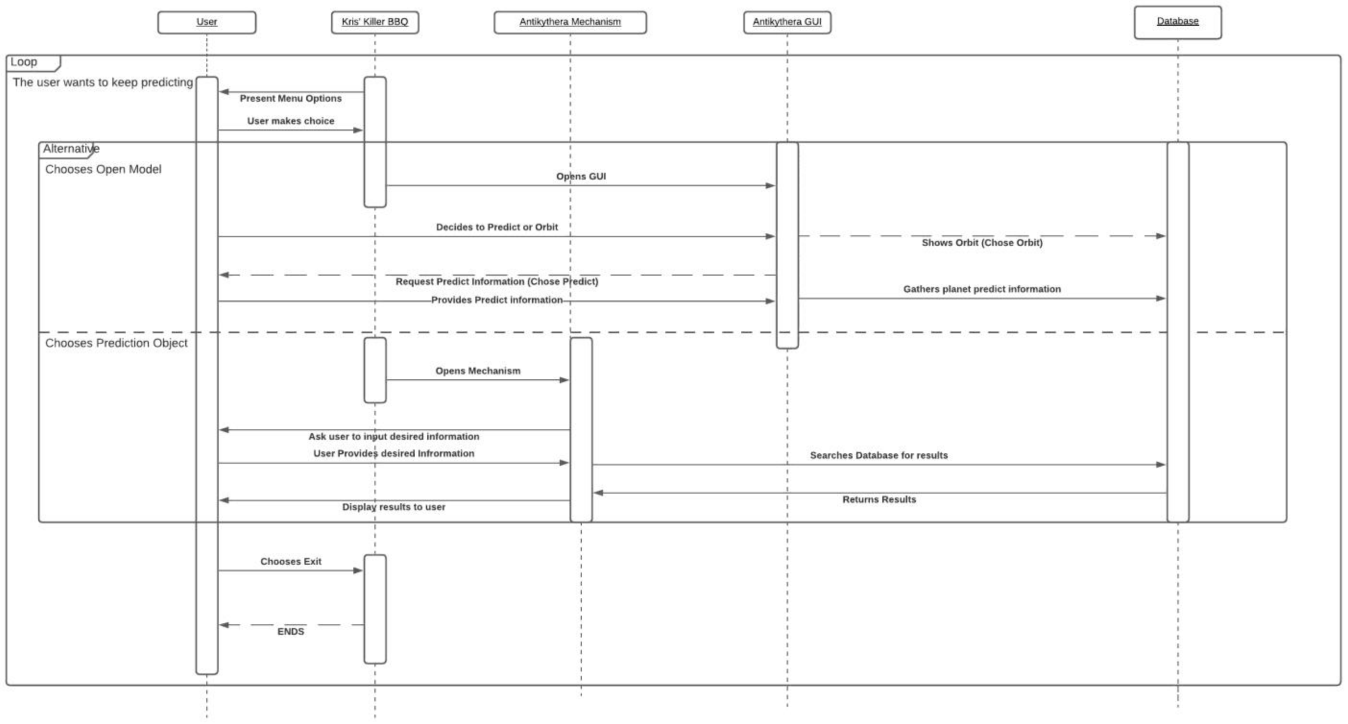
Use-Case UML


Planet Tracking an event prediction
Kris's BBQ
Antikythera Mech.
Antikythera GUI
Database
User
Choose Search Cagtegory
Verify Valid menu option
Display invalid input error
Call one of the other exe. based on menu decision or exit
Prompted to enter search information
Alternate Executable opens
Types in event prediction information
Decides to predict planets or track orbit
Passes search information to database
Display event prediction results to user
Displays planet results to user
Provides search results
{ Include }
{ Extend }
{ Extend }
{ Include }
{ Extend }
{ Include }
{ Extend }
{ Include }
{ Extend }
{ EXtend }

Activity UML


Start Program
Present Menu Options
Open Model
Open Antikythera GUI
Event Choice
Exit
Open Antikythera Mechanism
Planet Orbit
Select Orbit
Planets speeds collected from Database
Planets Orbit at their calculated speed
Reset
You WILL get an error message if you do not reset the display after orbitting or predicting
Planet Prediction
Input Month, year, or both
Select Predict
Planet Data is collected
Calculatd planet positions are presented
Reset
Request desired search parameters
input Search Parameters
Search database for input
Database returns requested information
Information Presented to user

Sequence UML


User
Kris' Killer BBQ
Antikythera Mechanism
Antikythera GUI
Database
Loop
The user wants to keep predicting
Present Menu Options
User makes choice
Alternative
Chooses Open Model
Opens GUI
Decides to Predict or Orbit
Shows Orbit (Chose Orbit)
Request Predict Information (Chose Predict)
Provides Predict information
Gathers planet predict information
Chooses Prediction Object
Opens Mechanism
Ask user to input desired information
User Provides desired Infrormation
Searches Database for results
Returns Results
Display results to user
Chooses Exit
ENDS

User Manual Summary

To begin running the Replicate Antikythera program, the files must be located in the main branch of the Replicate Antikythera GitHub repository. Begin by downloading the “ReplicateAntikythera.zip” folder and extract it to a folder located on your machine. Be sure to keep track of where the folder was extracted to.

Also note that within the downloaded folder is another folder named “bin”. Navigate to the location of the extracted folder and find the “bin” folder. Then, right click one of the executables inside the folder (such as c++filt.exe) and click “properties”. Once the “properties” window pops up, copy the file path from the “location” field, being sure to copy the entire path.

Once copied, close out of the “properties” window. Then, open your machine’s System Variables by opening Windows “Advanced System Settings” and from there, opening the “Environment Variables”. Once in “Environment Variables”, find the “Path” variable, click on it, and click “Edit”. From there, click “New” and then paste the file path you copied earlier from the “location” field. You can verify that it is correct by confirming that “bin\” is at the end.

Once pasted, click “Ok” and exit “Advanced System Settings”, and begin running “AntikytheraMechanism”.

Future work for Completion

While vastly modernizing this 2000+ year old mechanism, we believe there could be some quality-of-life improvements that could make it easier for users to get even more detailed information about celestial movements. The benefit with working with zero gravity is that it is easier to simulate and predict a body in space than it is with gravity. This means that tracking minor bodies or events can be simulated instead of manually entered into our database. Eclipses can be calculated using the relative positioning of the sun moon and earth. Commets can be tracked using detailed orbital tracking. The issue applies to planet planets too since they do not orbit in a perfect circle. These are all things that would be nice to implement so that that a user can use this program at any time in any year and still receive accurate results.

The GUI could use some improvement as well. Upon testing, users requested being able to select month and year for detailed planet positioning. It would also be nice to include an orbit feature from said positioning. The issue is, when simulating the position using the GUI, the orbit would consider that as the rotational origin hence why you need to reset the planets back to its default alignment to get accurate simulation. We would love to also package these two programs together, so they work simultaneously where one program handles the backend calculations while the GUI displays the results but unfortunately, we did not have enough time or knowledge to create a custom wrapper that is needed to handle the two languages.