LAP Visual IDE

Limitless Automation Possibilities

Team Info

Andrew

- Gym enjoyer
- Game Maker

Will

- Just another broke college senior
- Javascript flavors aficionado

Jacob

- Big gamer
- Can’t wait for semester to be over

Alex

- Music lover
- Programming is “fun”

Wesley

- Gamer (shocking)
- Coding is a lifestyle

Client Info

Blue Ridge Automation

- Develops automation software designed to make monotonous tasks easier
- Notable clients include:
  - Nestle
  - Kroger
  - Ingredion
  - Starbucks
- LAP is their main software that is sold
  - This is what our IDE will be for
  - Helps programmers develop automation code for factories visually and intuitively.

Mentor Feedback

Mid-iteration

- Looks good
- Stay focused on tasks that you have
- Communicate status of projects
- Told us to eat, sleep, breathe winning on the mind

Pre-release

- Communication of the status of tasks should be better
- Tasks shouldn’t be open for days
- Looks good
- Lots to look forward to next iteration

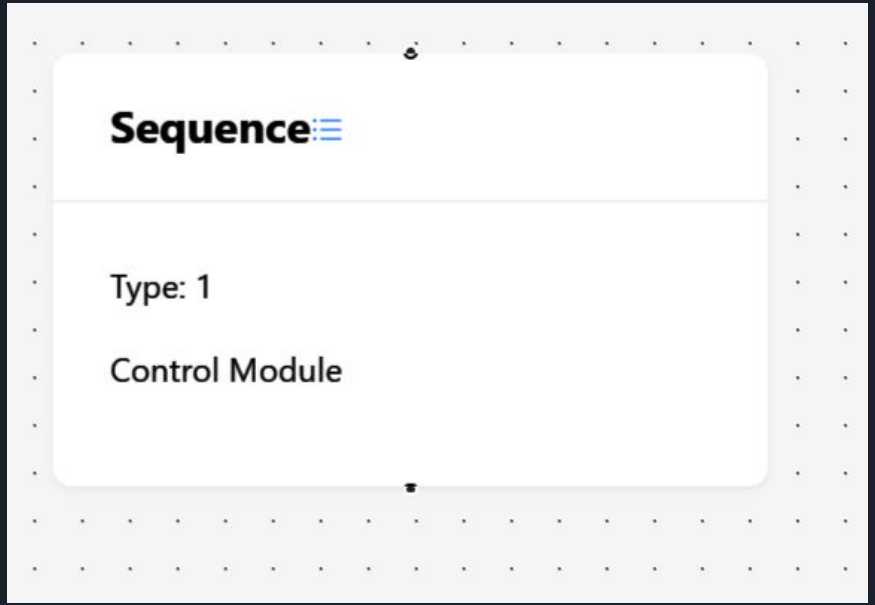
Client Feedback

- Program looks good and good progress has been made
- Some of the features were unneeded
    - Domain knowledge made it so that we had incorrectly implemented control modules, this has been fixed
- Clarified iteration 3 tasks
- Identified ways to make it more user friendly
    - Cache node color and type for next added node
- Identified ways to make the program more visually appealing

Iteration 2 Features

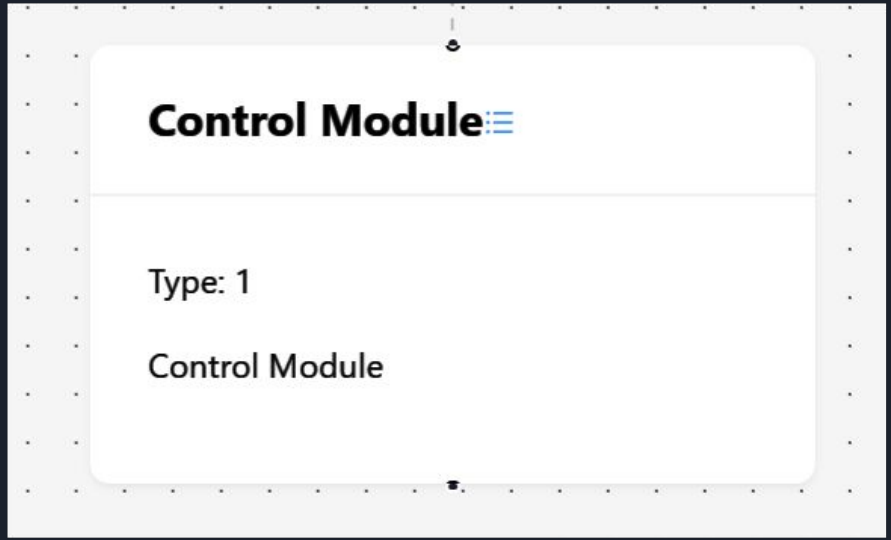
FR13: Custom Sequence

[FR13] There is a custom node type that represents a sequence node. [BR1] [HIGH]



FR19: Custom Control Module

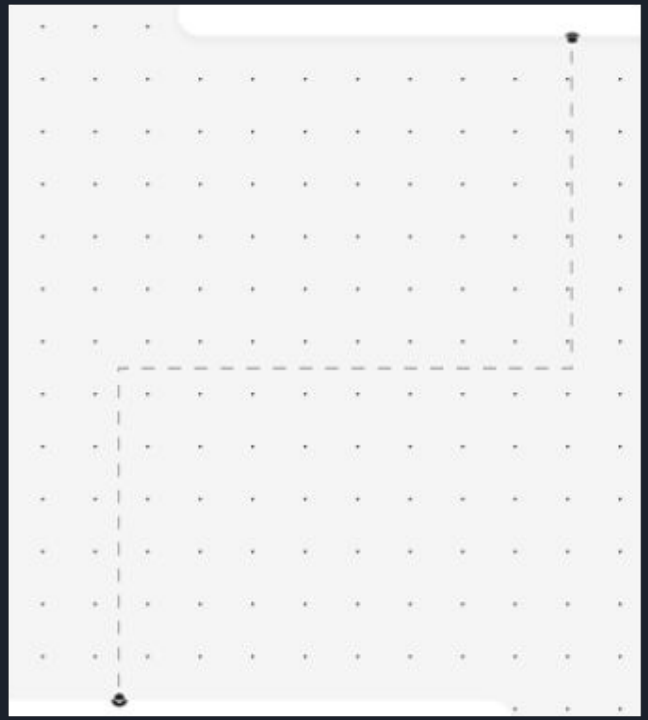
[FR19] There is a custom node type that represents a control module [BR1] [HIGH]



FR20: Line Direction

[FR20] The connection line between two nodes will show the direction of the connection[BR1][HIGH]

Imagine the lines are animated



FR17: Rest API

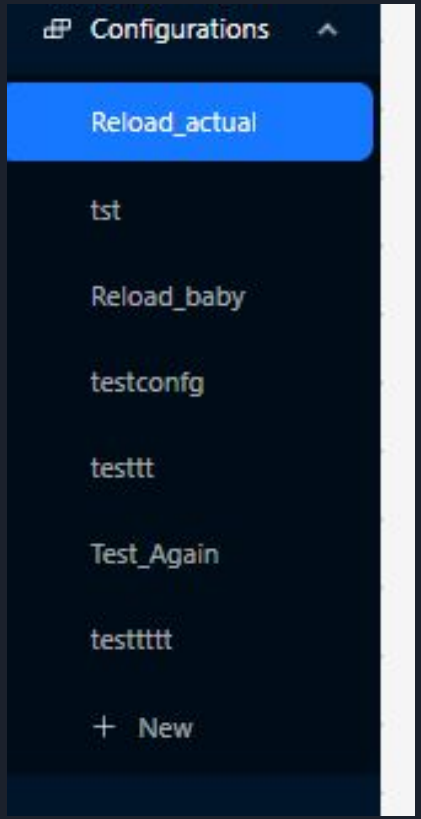
[FR17] A REST API will run alongside the application to push and pull the json to and from the database [BR1][HIGH]

I didn't know how else to show this

```
app.use(function (req, res, next) {
  res.header("Access-Control-Allow-Origin", "*");
  res.header(
    "Access-Control-Allow-Headers",
    "Origin, X-Requested-With, Content-Type, Accept"
  );
  next();
});

app.get("/initTest", (request, response) => {
  response.statusCode = 200;
  response.json({ info: "Request Received" });
});
//get queries
app.get("/getConfigJSON:cid", queries.getConfigurationData);
app.get("/getAllConfigs", queries.getAllConfigurations);
app.get("/getAllConfigsTest", queries.getAllConfigurationsTest);
app.get("/getConfigurationDataTest", queries.getConfigurationDataTest);

//destructiveQueries
app.post("/updateConfig", destructive.updateConfigurationData);
app.post("/insertNewConfig", destructive.insertNewConfiguration);
app.post("/updateConfigTest", destructive.updateConfigurationDataTest);
app.post("/insertNewConfigTest", destructive.insertNewConfigurationTest);

app.listen(port, () => {
  console.log(`App running on port ${port}.`);
});
```

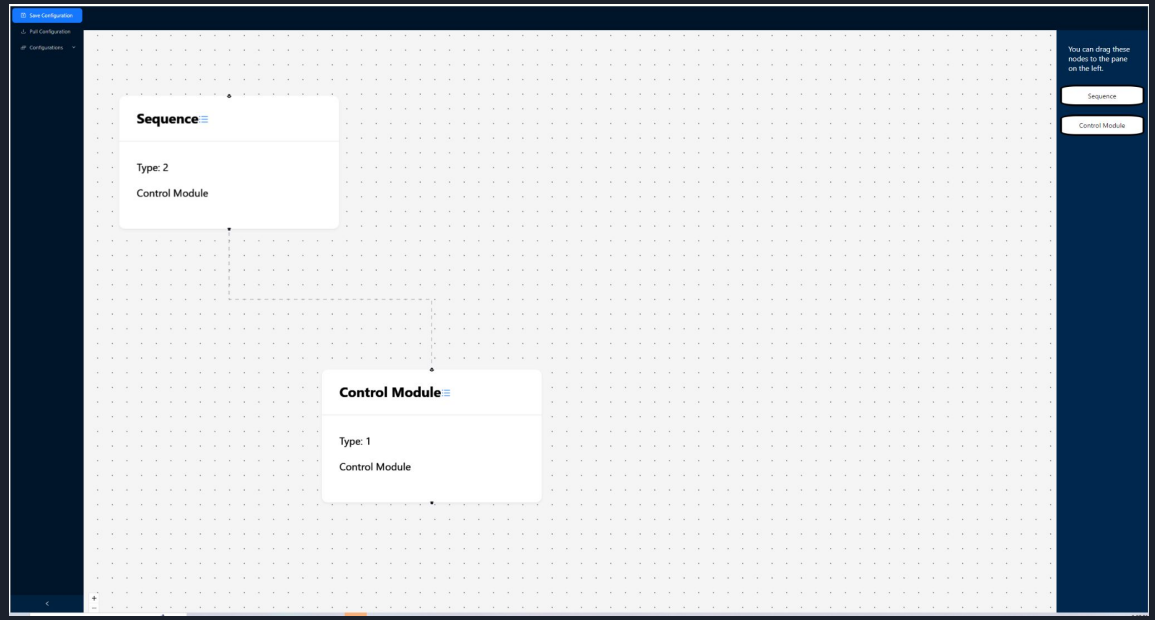
FR18: Edit From Database

[FR18] The configuration to edit will be pulled from the database. [BR1][MEDIUM]



NR2: Workspace Responsiveness

[NR2] The workspace should be responsive and intuitive to use. [BR2][LOW]



Planned Iteration 3 Features

1. User can access configurable settings through a menu on the step node
2. Each step has a pointer to another step or sequence
3. In the step configuration workspace, there will be a conditional block for every layer of it's parent sequence
4. Upon clicking the "configure steps" menu item, a new tab will open containing the workspace for that specific sequence
5. Each sequence node should visually have a menu item to configure the steps and subsequences of that sequence
6. The steps will be distinguished from each other through colors

Random Slide