Lab Conversion Template v2

LCT-v2

Nick Bennett, Salina Nihalani, Sebastian Wilson

Introduction

The LCT v2 Team

Nick Bennett
- 4th year Computer Science Major
- Skills: Java, Python, SQL, React/Javascript, C

Salina Nihalani
- 2nd year Computer Science Major
- Skills: Java, React/Javascript, Python

Sebastian Wilson

- 1st year Computer Science Major
- Skills: Java, JavaScript/React

What is LCT?

- Standard process for converting older MATLAB GUIs to JavaScript
- Using React
- Repository of functional code snippets
- Allows for hierarchical structure and capturing of state information

Why did we create LCT?

- Students have run into common problems every semester when creating Lab GUIs
- Having a standard process can avoid running into these problems repeatedly
- Allows students to focus on more complex graphing logic
- Overall will increase productivity for new students

What's new in LCT v2

- Restructured to take better advantage of React state handling
- Designed to include lab worksheets and GUIs together
- Improved state handling and saving/restoring of worksheet state
- Database connectivity

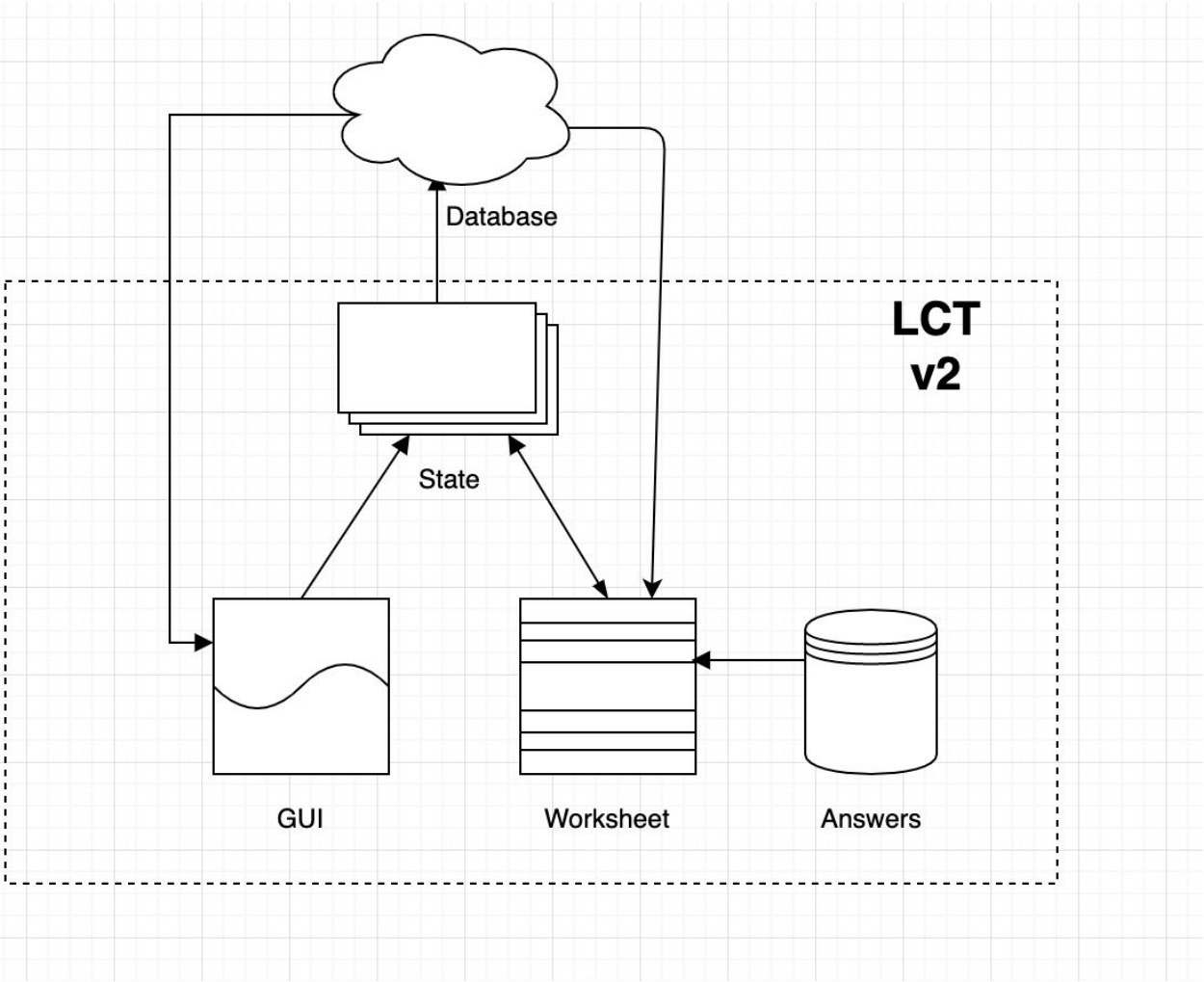
Project Design

Project Details

- Two demo apps: one frontend and one frontend + backend
- Created using create-react-app, jsxgraph-react-js, and express
- Uses React data model to share state information between GUI and Worksheet portions of the Lab
- Express backend for database connectivity

Why React?

- Built-in handling of data flow through states
- Modularity and reusability via Components
  - Allows for scalability along with simplicity
- Formal organization and structure
- Thriving ecosystem for packages and tools


Database
LCT
v2
State
GUI
Worksheet
Answers

Demo 1 (JSON Database)

- Restructured to include Worksheet and GUI sections
- Worksheet is able to read GUI state
- GUI is improved version of the LCT Test page from previous semester
- Contains implementations of every basic feature needed to create a GUI
- Code files are heavily annotated and organized to be easily copied/pasted into other projects
- Code structure more modular allowing for easy insertion of components

Worksheet Creation Template (Demo 1)

- Generates worksheet from static json file (WorksheetData.json)
- Predefined json file structure
- Defined by hierarchy of segments (Parts, questions, question parts, inputs)
- Standardized worksheet format
- Can interact with GUI state
- Modularity: Worksheet can have arbitrary segmentation with minimal hard coding

Worksheet Creation Template: Implementation (Demo 1)

- Constructor: Paste WorksheetData.json file contents and convert to new json file called “DynamicAnswers.json” which stores inputted data. In the state, either paste WorksheetData.json contents again or read from “DynamicAnswers.json.”
- createWorksheet(): Function that reads from WorksheetData, the static json file containing the worksheet questions, to generate arbitrary numbers of parts, questions, question parts, and input boxes, each with their own associated text.
- handleInputChange(): Updates “DynamicAnswers.json”

Worksheet Creation Template: JSON Structure

```
"Parts": [
    {
        "part": "Part identification number, eg. 1",
        "header": "Header/title for entire part ,eg. 2 Pre-Lab",
        "introText": "Below header, introduces part, eg. In this part, you will be...",
        "questions": [{
            "question": "Question number, eg. 2.1",
            "header": "Header title for question, eg. 2.1 Sampling an Aliasing GUI",
            "introText": "Text introducing question, eg. The first objective of this lab is..."
            "questionParts": [{
                "questionPart": "Question part identification letter, eg. a",
                "text": "Text introducing question part, eg. (a) Set the input to...",
                "inputs": [{
                    "text": "Text before input box, eg. Enter your first number here: ",
                    "responseType": "Expected data type of input, eg. number"
        "answers": [{
                 "answer": "question id number, eg. 2.1",
                 "answerParts": [{
                         "answerPart": "question part id letter, eg. a",
                         "correctAnswers": ["Correct answer, eg. 1"],
                         "inputtedAnswers": ["Student inputs are stored here, initially empty"]
```

Demo Two (MySQL database)

- Utilizes MySQL database to store worksheet components in a table
- Secures a connection to database using TypeScript
- Frontend utilizes React to display basic worksheet

Next Steps

- Fully integrate database and backend
- Finish implementation of real labs
- Finish worksheet generation to be more robust
- Integrate more reusability via React Components
- Gather feedback