APIs and Agents Part 2

CS 691 / SWE 699
Fall 2025

GEORGE MASON UNIVERSITY

Logistics

- Lecture 4 reading questions due today at 4:30pm
- Lecture 5 reading questions for next week due 9/25 at 4:30pm
- Reflection 2: Due 9/25 at 4:30pm

Today

- In-Class Activity
  - Build a design and architecture exploration tool
- HW (due next week before class)
  - Reflection due on Lecture 2 activity

In-Class Activity

- Goal today: see how far you can go to build a real world web app built on top of APIs
- Not important to finish, but want to push limits on what you can do and see where that breaks down
- Use Kaltura to record your screen for all of your programming work, upload video at end of class
- HW assignment (due next week) will ask you to reflect on your experiences today in class, and will be helpful to go back and reference your video

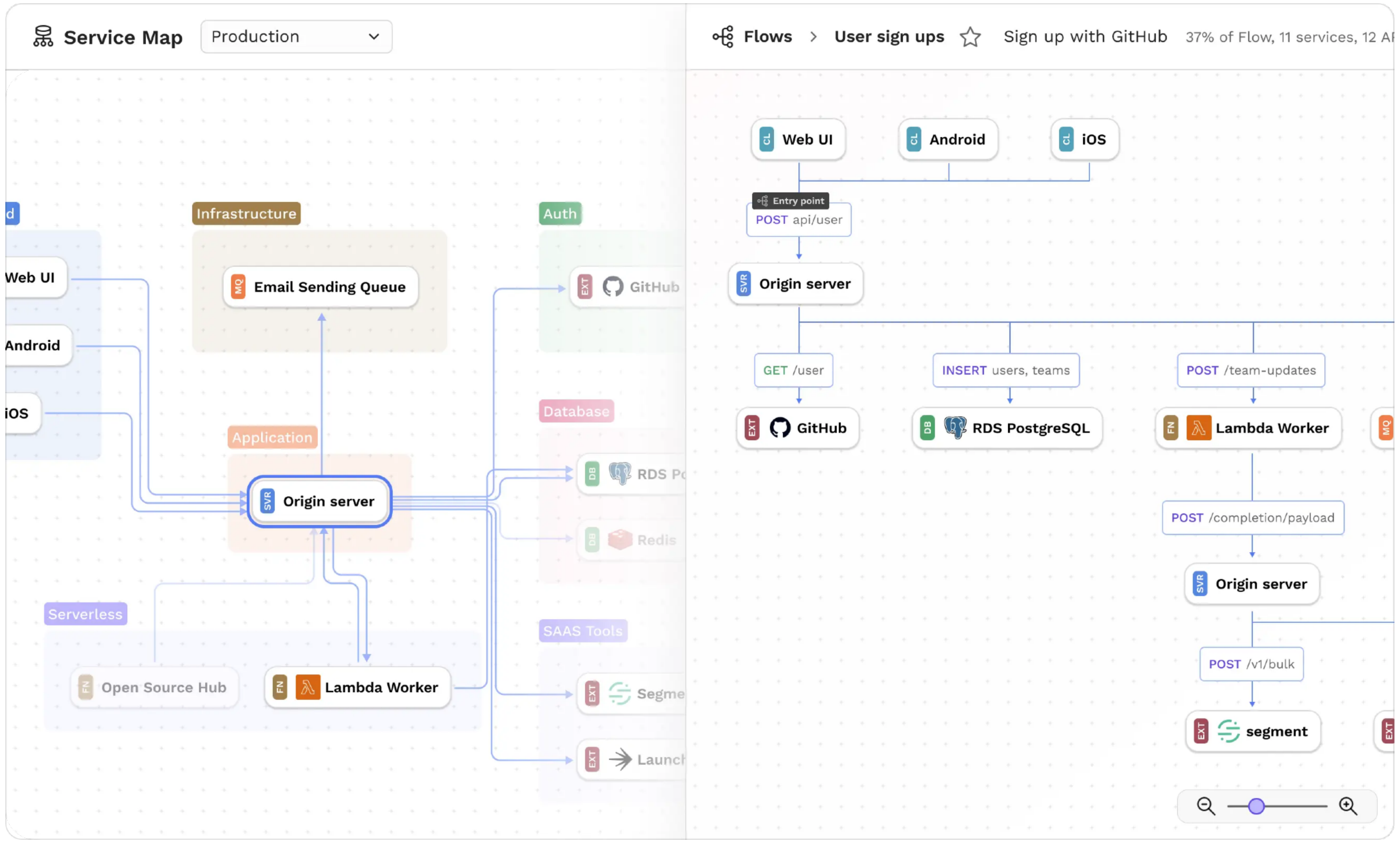
Cleaning Vibe-Coded Code

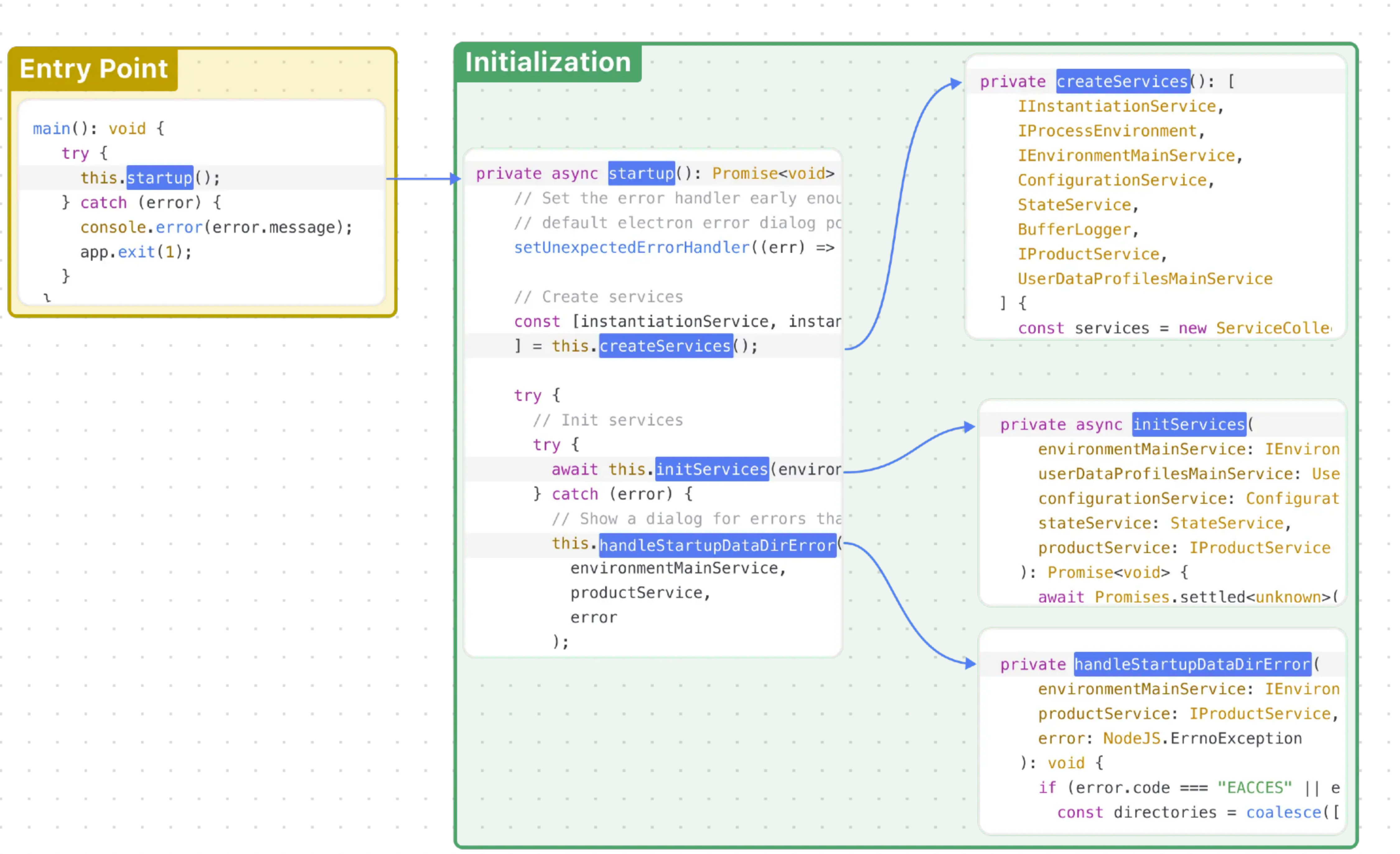
- Vibe coding can create messy code which is hard to understand and maintain
- Goal: build a developer in the loop web app that helps the developer understand the current design & architecture of code, identify design & code quality problems, and prompt coding agent (e.g., Cursor) to fix
    - Use APIs along the way for core functionality
- Input: root directory of a project repo (e.g., your City Simulation from Lecture 2)
- Outputs: a web app that *interactively* visualizes design & architecture of code, helping find issues
        - prompts that can be used by LLM to improve the design of the code

Suggested List of Features

1. Extract textual summary of architectural & design facts about code
2. Build diagram that communicates design facts about codebase, letting developers see relationships between packages, classes, & files (could use d3.js or build directly in svg)
3. Let developers drill into elements in diagram to see code inside, overlaid on diagram
4. Let developers see control flow between methods, starting from specific method in codebase, overlaid on diagram
5. Use a code clone algorithm (or reuse clone detection library) to find code duplication, & highlight to user in diagram
6. Use smell detectors, or other code quality assessments, to find & show issues with code
7. Enable developers to propose high-level refactorings to reduce code duplication, increase maintainability, and cleanup code
8. Build MCP interface for tool: let Cursor use MCP to invoke tool, halting execution while developer interacts with tool, and outputting a refactoring prompt to have Cursor execute
9. [Propose your own feature, explaining how it helps achieve project goal]

Examples of visualizations





```
main(): void {
  try {
    this.startup();
  } catch (error) {
    console.error(error.message);
    app.exit(1);
  }
```

```
private async startup(): Promise<void>
    // Set the error handler early enou
    // default electron error dialog pc
    setUnexpectedErrorHandler((err) =>

    // Create services
    const [instantiationService, instar
    ] = this.createServices();

    try {
      // Init services
      try {
        await this.initServices(enviror
      } catch (error) {
        // Show a dialog for errors tha
        this.handleStartupDataDirError(
          environmentMainService,
          productService,
          error
        );
```

```
private createServices(): [
    IInstantiationService,
    IProcessEnvironment,
    IEnvironmentMainService,
    ConfigurationService,
    StateService,
    BufferLogger,
    IProductService,
    UserDataProfilesMainService
  ] {
    const services = new ServiceColle
```

```
private async initServices(
    environmentMainService: IEnviron
    userDataProfilesMainService: Use
    configurationService: Configurat
    stateService: StateService,
    productService: IProductService
  ): Promise<void> {
    await Promises.settled<unknown>(
```

```
private handleStartupDataDirError(
    environmentMainService: IEnviron
    productService: IProductService,
    error: NodeJS.ErrnoException
  ): void {
    if (error.code === "EACCES" || e
      const directories = coalesce([
```

Extra Credit

Extra

- 10 points extra credit
  - Get at least 100 stars on a public GitHub repo, illustrating real world use & interest in your tool

In-Class Activity

- Have from now till end of class period to work (7:10pm)
- Share video at end of class
    - Upload your video to your own OneDrive space
    - Include a link in your reflection (make sure visibility set to make it viewable)