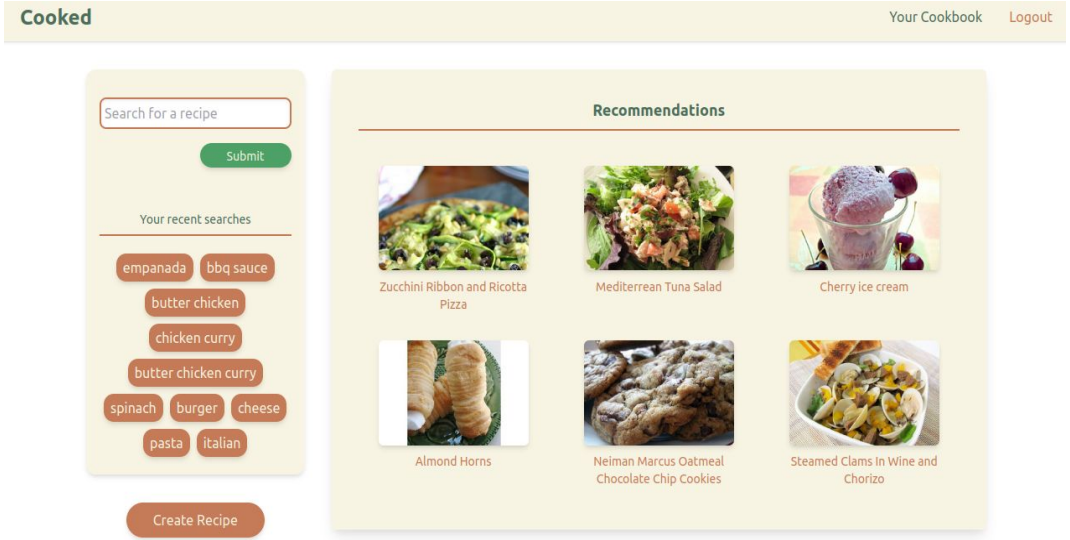
Cooked Presentation
NWEN304 Group Project

Ahad Rahman
Daniel Pullon
Eris Atienza
August Bolter

Introduction

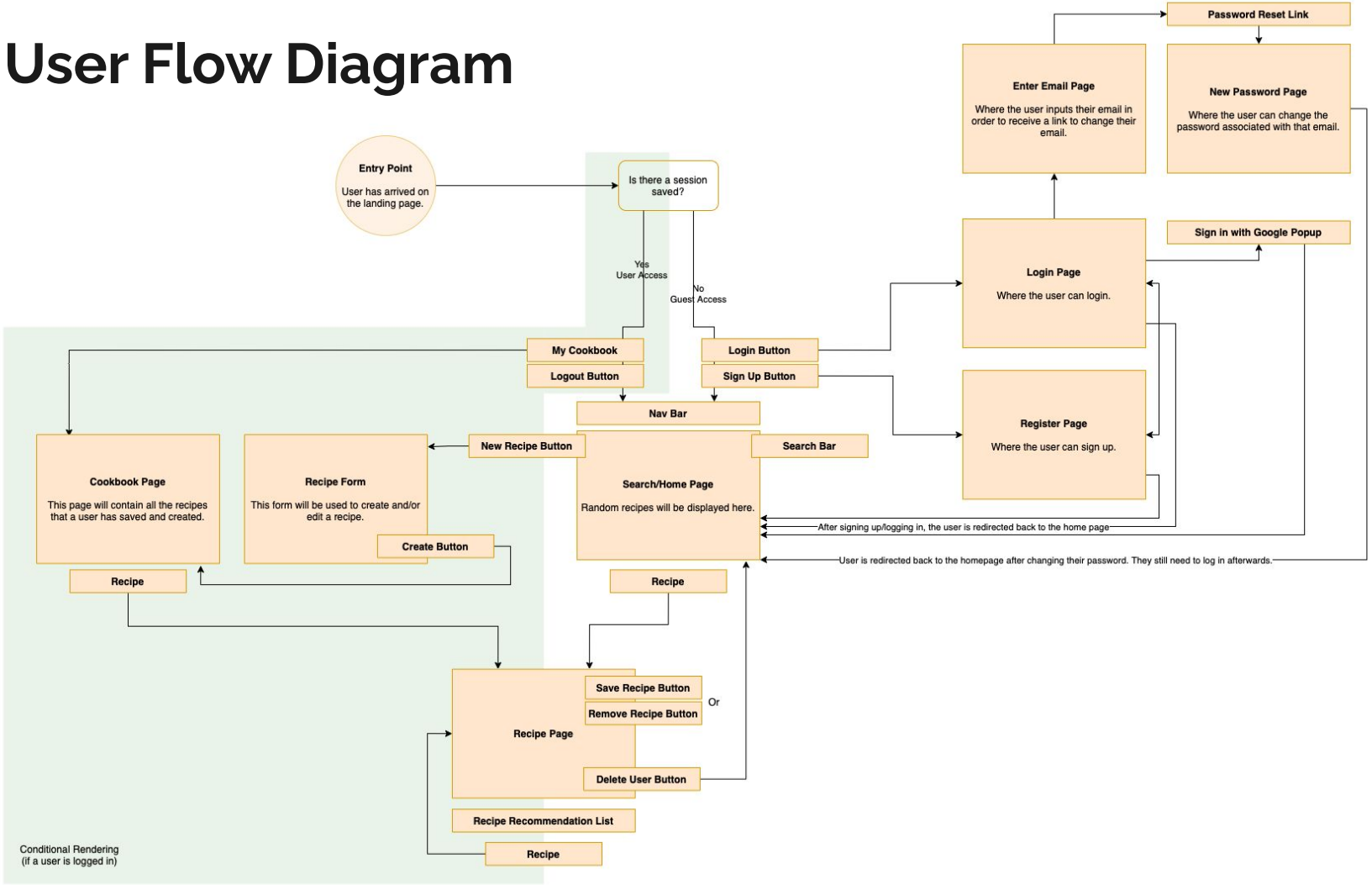
- Cooked is a website to create and save recipes
- Search thousands of recipes
- View saved recipes within your own cookbook
- Receive custom recommendations based on the recipes you search for
- Your account is secure and can be accessed from everywhere



High-Level Design

- Follows the Model-View-Controller software design pattern
- Web service and front end web app are separated; exist independently from each other
- CRUD operations
- Uses a NoSQL data store; MongoDB
- Two types of users:
  - Guest -> Basic app functionality; view and search for recipes.
  - Registered User -> Able to write and save recipes to their cookbook, save their recent searches, and use a recommendation system based on the recipe they're viewing.
- App's behaviour differs whether the user is registered or not
  - Limited functionality, conditional rendering of components

User Flow Diagram



Contributions: Eris

- App Planning
  - What is our app, what does it do, and who are the users?
  - Model: table schemas
  - View: wireframes, and user flow diagram
  - Controller: potential end-points
- GET requests from Spoonacular API for recipe summary and similar recipes
- Created Recipe Model
- Front-end
  - Entire styling of the app
  - Some of the page routing
  - Some of the view and controller integration
- Delete User Functionality
- Password Reset Functionality

Contributions: Ahad

- Express routing on both web service and web app front end
- Communication with Spoonacular API from web service
- EJS setup
- Tailwind (CSS framework) setup
- 404 page
- Search functionality with routing to recipe details page
- MVC modelling (directory structure)
- Display recent searches
- Recommending similar recipes on recipe details page
- OAuth/Open ID Authentication (Sign in with Google)
- Performance Testing

Contributions: Daniel

- MongoDB Atlas setup and NoSQL DB design
- Server design
- Many of the CRUD functions including:
  - Creating cookbooks
  - Getting recipes
  - Deleting cookbooks
  - Saving user searches
- API calls from the client to the server
- Setting up hosting on Heroku

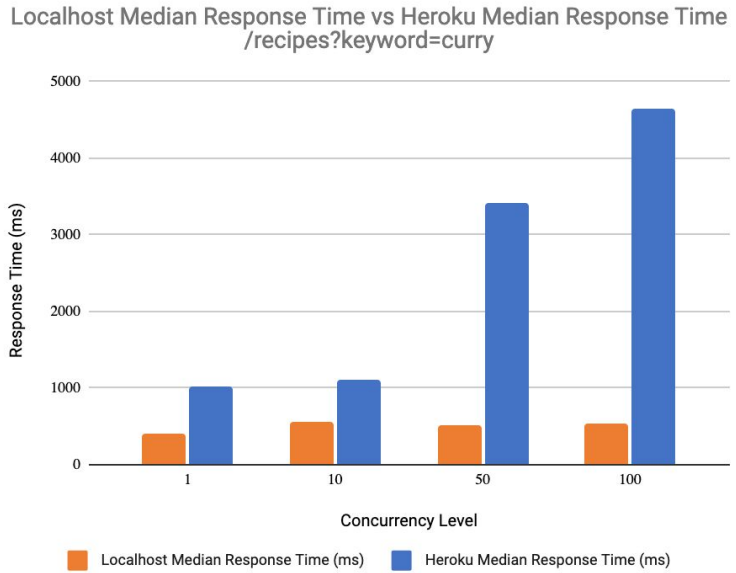
Contributions: August

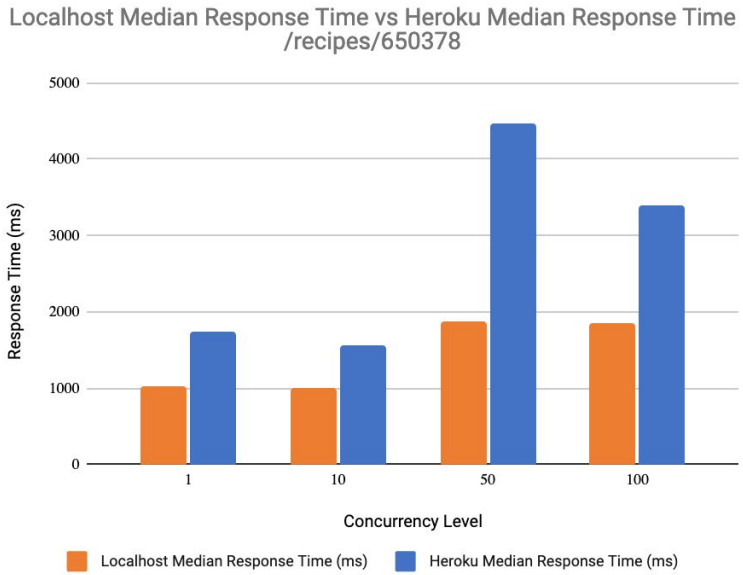
- Registration and logging in
- JWT authorisation
- Password reset functionality
- Inactivity timer
- Create Recipe page and functionality
- Cookbook page and functionality
- Save recipe and remove recipe functionality
- MVC modelling (directory structure)

Performance Analysis

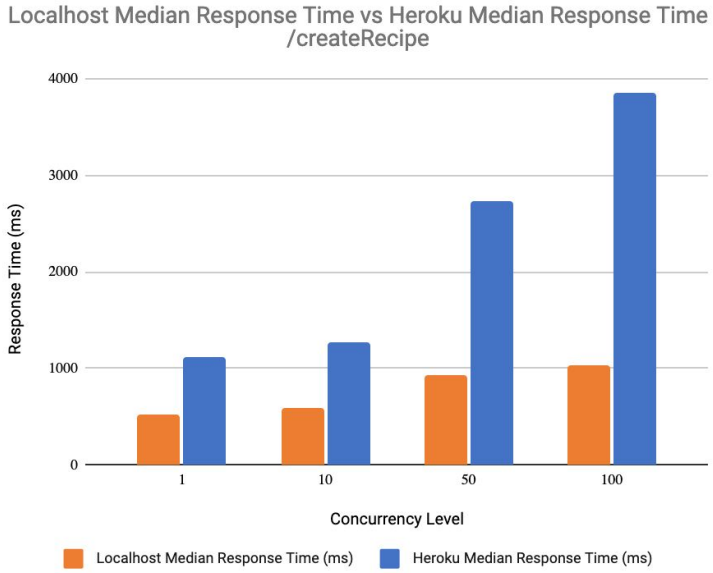
- For each test, we send 100 requests with varying levels of concurrency to both localhost and heroku
- We measured the median response time of one request

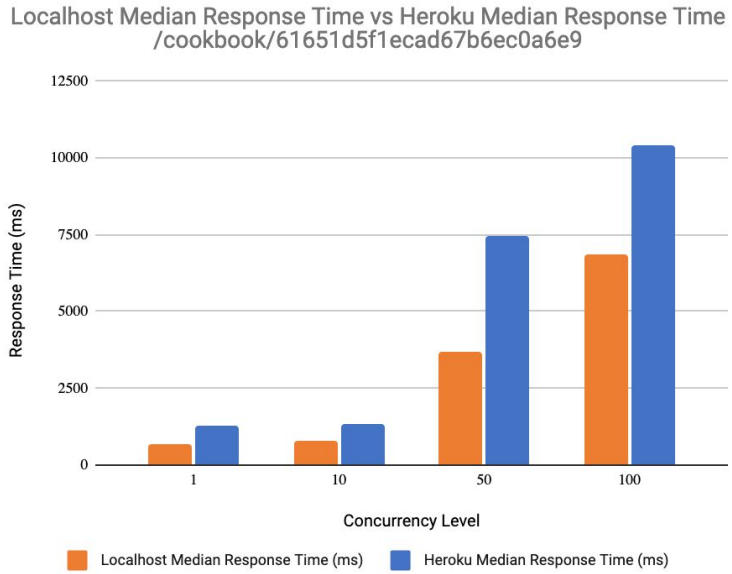
Performance Analysis





Performance Analysis





Evaluation of performance of database

- MongoDB Free Tier
- Little performance limitations, parts of UI that relied on DB calls would render slightly later
- Maximum number of connections allowed on free tier was not enough for more rigorous testing
- Response time increased with number of concurrent connections
- Some bottlenecking
- Minimal during development
- Paid plans to scale database both horizontally and vertically

Thank you!